– EPE –
NEW NAME
NEW DESIGN!

Practical Electronics

The UK's premier electronics and computing maker magazine

Micromite
Subroutines and interrupts

Circuit Surgery
Circuit simulation with LTspice

Cool Beans
'Micing' an Arduino

PIC n' Mix
Small, cheap and powerful PICs

Wide-range Arduino-based LC Meter

L:95.63uH 372051Hz
C:0.000 F 707590Hz
--> Inductor
L=95.559uH 80 samp

WIN!
LAN7430 PCIe 3.1 Ethernet Bridge Evaluation Board

USB Flexitimer

Build your own Theremin!

MIDI Ultimate Synthesiser – Part 5

Putting MIDI in the MIDI Ultimate

PLUS!

4GHz Digital Attenuator

Net Work, Techno Talk and Electronic Building Blocks

TEACH-IN 2019 – Powering Electronics

June 2019 £4.65
06
9 770262 361195

www.epemag.com | @practicalelec | practicalelectronics

With Over 250 Million Units in Stock, We've Got You Covered

Delivering You the World's Largest Inventory of Microchip Products

www.microchip.com/learn-more

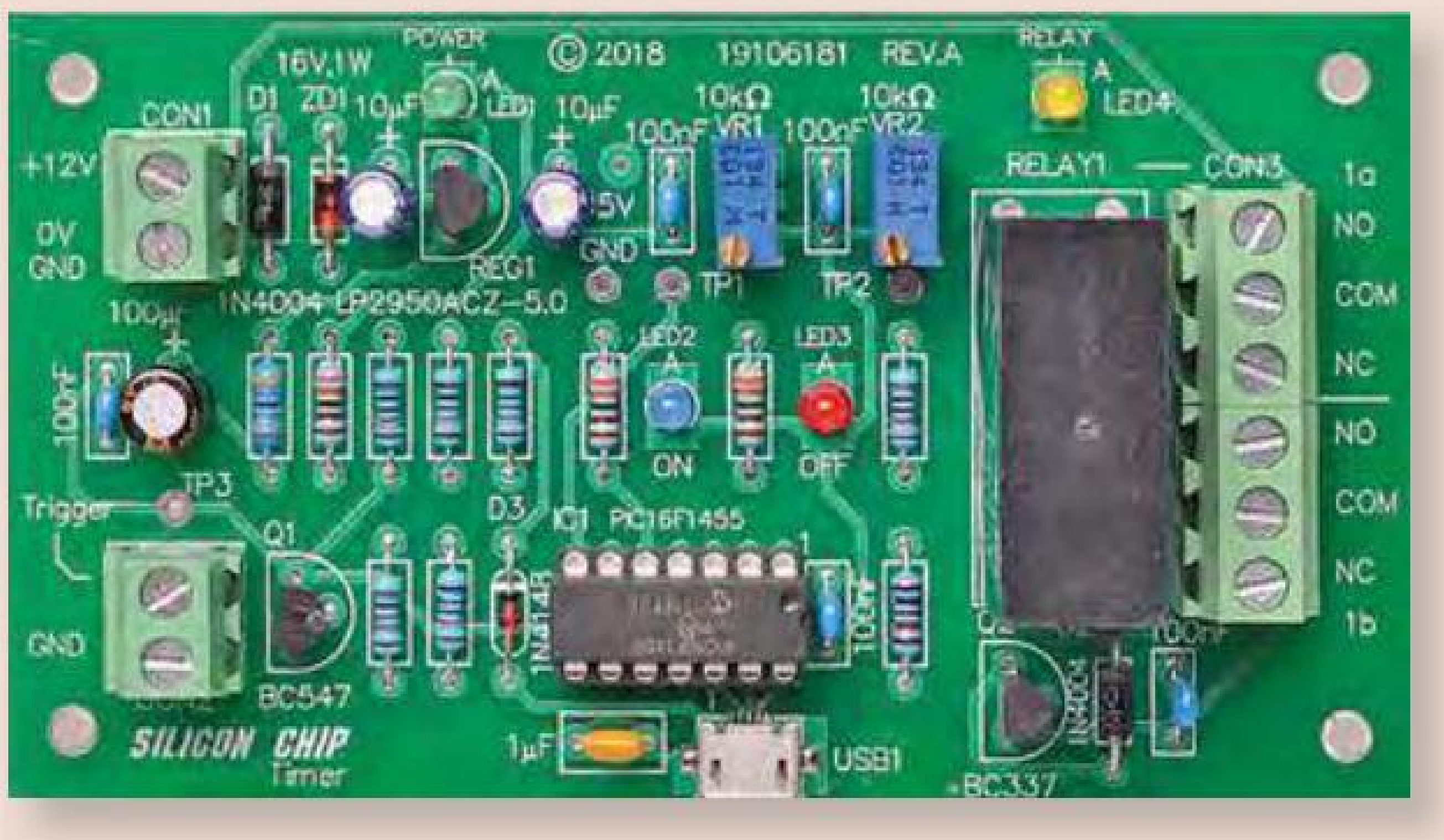
MICROCHIP

The Microchip name and logo and the Microchip logo are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.
All other trademarks are the property of their registered owners.
© 2018 Microchip Technology Inc. All rights reserved. DS00002552A. MEC2226Eng10/18

Practical Electronics

Volume 48. No. 6
June 2019
ISSN 0262 3617

Contents



© Electron Publishing Limited 2019

Copyright in all drawings, photographs, articles, technical designs, software and intellectual property published in *Practical Electronics* is fully protected, and reproduction or imitation in whole or in part are expressly forbidden.

The July 2019 issue of *Practical Electronics* will be published on Thursday, 6 June 2019 – see page 80.

Projects and Circuits

Wide-range Arduino-based LC Meter by Tim Blythman 14
This very low-drift, self-calibrating digital inductance-capacitance meter is easy to build – it's based on a custom Arduino shield and standard 4-line LCD display.

USB Flexitimer by John Clarke and Nicholas Vinen 24
A very flexible timer. It switches its relay at intervals from milliseconds up to many days; plus, you can fully configure it with your laptop PC via USB.

Using Cheap Asian Electronic Modules – Part 17 by Jim Rowe 32
Learn to use a highly linear 4GHz digital attenuator programmable over 0dB to –31.5dB in 0.5dB steps.

MIDI Ultimate Synthesiser – Part 5 by Paul Cooper 36
Overview of the Musical Instrument Digital Interface (MIDI) and how it is implemented within the MIDI Ultimate synth.

Series, Features and Columns

News 8
A roundup of the latest news from the world of electronics and technology

Techno Talk by Mark Nelson 10
New Light on LEDs

Net Work by Alan Winstanley 12
Updating, extending and (hopefully) simplifying home-networked Internet access

Teach-In 2019 – Powering Electronics by Mike Tooley 46
Part 7: Heatsinks and negative outputs

Audio Out by Jake Rothman 52
The PE Theremin – Part 1

Circuit Surgery by Ian Bell 56
Introduction to Circuit Simulation with LTspice – Part 5

PIC n' MIX by Mike Hibbett 60
Small, cheap and powerful – Part 2

Max's Cool Beans by Max The Magnificent 65
Options for 'micing' an Arduino

Make it with Micromite by Phil Boyce 66
Part 5: Traffic lights, subroutines, interrupts and IR control

Electronic Building Blocks by Julian Edgar 76
Thermal imaging cameras

Regulars and Services

Subscribe to Practical Electronics and save money 4
Practical Electronics back issues CD-ROM – great 15-year deal! 6
Reader services – Editorial and Advertising Departments 7
Editorial 7
50 years of progress
Exclusive Microchip reader offer 23
Win a LAN7430 PCIe 3.1 Ethernet Bridge Evaluation Board
Teach-In bundle – what a bargain! 31
Practical Electronics – get your back issues here! 64
Practical Electronics CD-ROMS for electronics 70
A superb range of CD-ROMs for hobbyists, students and engineers
Direct Book Service 73
Build your library of carefully chosen technical books
Practical Electronics PCB Service 78
PCBs for Practical Electronic projects
Advertiser index 79
Next month! – highlights of our next issue of Practical Electronics 80

QUASAR
electronics

Quasar Electronics Limited
PO Box 6935, Bishops Stortford
CM23 4WP, United Kingdom

Tel: 01279 467799
E-mail: sales@quasarelectronics.co.uk
Web: quasarelectronics.co.uk

All prices INCLUDE 20.0% VAT. Free UK delivery on orders over £48
Postage & Packing Options (Up to 0.5Kg gross weight): **UK Standard 3-7 Day Delivery** - £4.95; **UK Mainland Next Day Delivery** - £9.95; **Europe (EU)** - £12.95; **Rest of World** - £14.95 (up to 0.5Kg).
!! Order online for reduced price postage and fast despatch !!
Payment: **We accept all major credit/debit cards. Make cheques/PO's payable to Quasar Electronics Limited.**
Please visit our online shop now for full details of over 1000 electronic kits, projects, modules and publications. Discounts for bulk quantities.

QuasarElectronics.co.uk

Solutions for Home, Education & Industry Since 1993

cebek
velleman

Official Main Dealer stocking the full range of Kits, Modules, Robots, Instruments, tools and much, much more...

PIC & ATMEL Programmers

We have a wide range of PIC, ATMEL Arduino and Raspberry Pi projects.

PIC Programmer & Experimenter Board

Great learning tool. Includes programming examples and a reprogrammable 16F627 Flash Microcontroller. Test buttons & LED indicators. Software to compile & program your source code is included. Supply: 12-15Vdc. Pre-assembled and ready to use.
Order Code: VM111 - ***~~£38.88~~*** ***£35.94***

USB PIC Programmer and Tutor Board

The only tutorial project board you need to take your first steps into Microchip PIC programming using a PIC16F882 (included). Later you can use it for more advanced programming. Programs all the devices a Microchip PICKIT2® can! Use the free Microchip tools for PICKit2™ & MPLAB® IDE environment.
Order Code: EDU10 - ***£46.74***

USB /Serial Port PIC Programmer

Fast programming. Wide range of PICs supported (see website for details). Free Windows software & ICSP header cable. USB or Serial connection. ZIF Socket, leads, PSU not included.
Kit Order Code: 3149EKT - ***~~£49.96~~*** ***£29.95***
Assembled Order Code: AS3149E - ***£44.95***
Assembled with ZIF socket Order Code: AS3149EZIF - ***~~£74.96~~*** ***£49.95***

PICKit™2 USB PIC Programmer Module

Versatile, low cost, PICKit™2 Development Programmer. Programs all the devices a Microchip PICKIT2 programmer can. Onboard sockets & ICSP header. USB powered.
Assembled Order Code: VM203 - ***£35.94***

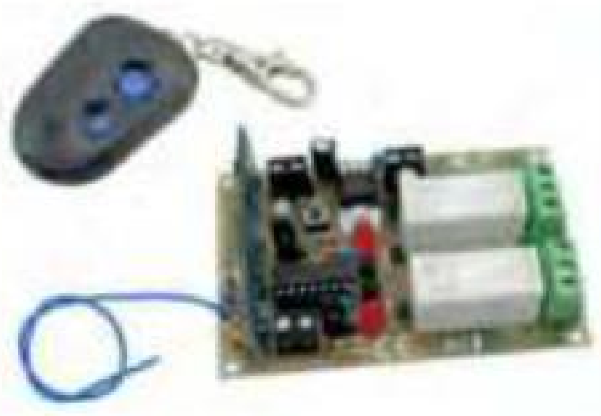
Controllers & Loggers

Here are just a few of the controller and data acquisition and control units we have. See website for full details. 12Vdc PSU for all units: Order Code 660.446UK £10.68

USB Experiment Interface Board

Updated Version! 5 digital inputs, 8 digital outputs plus two analogue inputs and two analogue outputs. 8 bit resolution. DLL.
Kit Order Code: K8055N - ***~~£39.95~~*** ***£22.20***
Assembled Order Code: VM110N - ***£35.94***

2-Channel High Current UHF RC Set



State-of-the-art high security. Momentary or latching relay outputs rated to switch up to 240Vac @ 12 Amps. Range up to 40m. 15 Tx's can be learnt by one Rx. Kit includes one Tx (more available separately). 9-15Vdc.
Kit Order Code: 8157KT - ***£44.95***
Assembled Order Code: AS8157 - ***£49.96***

Computer Temperature Data Logger

Serial port 4-ch temperature logger. °C/°F. Continuously log up to 4 sensors located 200m+ from board. Choice of free software applications downloads for storing/using data. PCB just 45x45mm. Powered by PC. Includes one DS18S20 sensor.
Kit Order Code: 3145KT - ***~~£19.95~~*** ***£16.97***
Assembled Order Code: AS3145 - ***£19.96***
Additional DS18S20 Sensors - ***£4.96 each***

8-Channel Ethernet Relay Card Module

Connect to your router with standard network cable. Operate the 8 relays or check the status of input from anywhere in world. Use almost any internet browser, even mobile devices. Email status reports, programmable timers... Test software & DLL online.
Assembled Order Code: VM201 - ***£130.80***

Computer Controlled / Standalone Unipolar Stepper Motor Driver

Drives any 5-35Vdc 5, 6 or 8-lead unipolar stepper motor rated up to 6 Amps. Provides speed and direction control. Operates in stand-alone or PC-controlled mode for CNC use. Connect up to six boards to a single parallel port. Board supply: 9Vdc. PCB: 80x50mm.
Kit Order Code: 3179KT - ***£15.26***
Assembled Order Code: AS3179 - ***£22.26***

Card Sales & Enquiries **01279 467799**

Bidirectional DC Motor Speed Controller

Control the speed of most common DC motors (rated up to 32Vdc/5A) in both the forward and reverse directions. The range of control is from fully OFF to fully ON in both directions. The direction and speed are controlled using a single potentiometer. Screw terminal block for connections. PCB: 90x42mm.
Kit Order Code: 3166KT - ***£19.99***
Assembled Order Code: AS3166 - ***£29.99***

8-Ch Serial Port Isolated I/O Relay Module

Computer controlled 8 channel relay board. 5A mains rated relay outputs and 4 opto-isolated digital inputs (for monitoring switch states, etc). Useful in a variety of control and sensing applications. Programmed via serial port (use our free Windows interface, terminal emulator or batch files). Serial cable can be up to 35m long. Includes plastic case 130x100x30mm. Power: 12Vdc, 500mA.
Kit Order Code: 3108KT - ***£74.95***
Assembled Order Code: AS3108 - ***£89.95***

8-Channel RF Remote Control Set

Control 8 onboard relays with included RF remote control unit. Toggle or momentary mode for each output. Up to 50m range. Board Supply: 12Vac, 500mA
Assembled Order Code: VM118 - ***£71.94***

Temperature Monitor & Relay Controller

Computer serial port temperature monitor & relay controller. Accepts up to four Dallas DS18S20 / DS18B20 digital thermometer sensors (1 included). Four relay outputs are independent of the sensors giving flexibility to setup the linkage any way you choose. Commands for reading temperature / controlling relays are simple text strings sent using a simple terminal or coms program (e.g. HyperTerminal) or our free Windows application. Supply: 12Vdc.
Kit Order Code: 3190KT - ***~~£79.96~~*** ***£47.95***
Assembled Order Code: AS3190 - ***£59.95***

3x5Amp RGB LED Controller with RS232

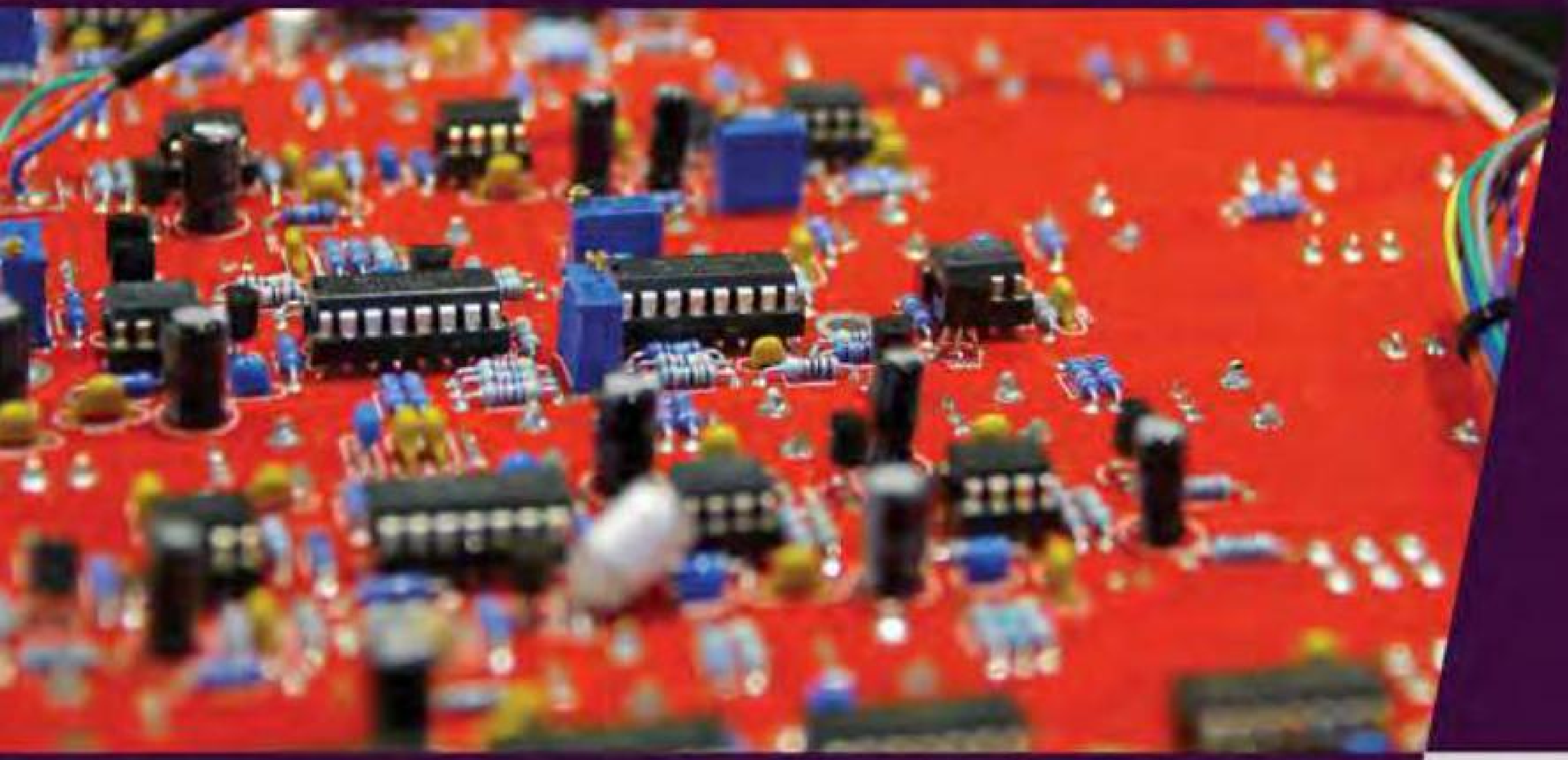
3 independent high power channels. Preprogrammed or user-editable light sequences. Standalone or 2-wire serial interface for microcontroller or PC communication with simple command set. Suits common anode RGB LED strips, LEDs, incandescent bulbs. 12A total max. Supply: 12Vdc. 69x56x18mm
Kit Order Code: 8191KT - ***£24.95***
Assembled Order Code: AS8191 - ***£27.95***

www.Soundtronics.co.uk

MIDI Ultimate Analogue Synth Kits

As Featured In Practical Electronics

60+ analogue synth module kits are available with our matching 5U format front panels, based on the renowned MFOS and YuSynth designs. Kits range from VCO's to 16 step analogue sequencers



A range of 5 single board synths available, each with essential modular synth functions. All built into a compact enclosure, including the popular MFOS Sound Lab Ultimate

Manufactured in-house, our attractive hardwood modular cabinets and desktop cases are a great way to complete your build. Available in maple, oak and walnut timbers.

MIDI Ultimate
- Parts & complete kits
- Front panels
- Hardwood case

Available at soundtronics

Over 1,000 products to choose from including kits, PCB's and individual components

facebook.com/SoundtronicsUK
www.soundtronics.co.uk

– EPE –
NEW NAME
NEW DESIGN!

Practical Electronics

UK readers SAVE 73p on every issue

Why not pay £3.92 instead of £4.65 for your copy of Practical Electronics?

Just take out a one-year subscription and save 73p on every issue, or £8.80 over the year.
Even better – save 94p per issue if you subscribe for two years – a total saving of £22.60

Overseas rates also represent exceptional value

You also:
- **Avoid any cover price increase for the duration of your subscription**
- **Get your magazine delivered to your door each month**
- **Guarantee a copy, even if the newsagents sell out**

Order by:
- **Phone or post with a cheque, postal order or credit card**
- **Or order online at: www.epemag.com (go to our secure server-backed online shop)**

Practical Electronics

SUBSCRIPTION PRICES

Subscriptions for delivery direct to any address in:

UK: 6-months £25.00, 12-months £47.00, 24-months £89.00

Europe Airmail: 6-months £30.00, 12-months £56.00, 24-months £107.00

Rest Of The World Airmail: 6-months £37.00, 12-months £70.00, 24-months £135.00

Cheques (in **£ sterling only**) payable to Practical Electronics and sent to: Wimborne Publishing Ltd, 113 Lynwood Drive, Merley, Wimborne, Dorset BH21 1UU.

Tel: 01202 880299. **Fax:** 01202 843233.
Email: stewart.kearn@wimborne.co.uk.

Also via our secure online shop at: **www.epemag.com.**

Subscriptions start with the next available issue. We accept MasterCard or Visa.
(For past issues see the Back Issues page.)

ONLINE SUBSCRIPTIONS

Electronic subscriptions (PDF download or online) are available, £22.00 for 12 months, visit **www.epemag.com** for more details.

SUBSCRIPTION ORDER FORM

☐ **6 Months: UK £25.00, Europe £30.00 (Airmail), Rest of the World £37.00 (Airmail)**
☐ **1 Year: UK £47.00, Europe £56.00 (Airmail), Rest of the World £70.00 (Airmail)**
☐ **2 Years: UK £89.00, Europe £107.00 (Airmail), Rest of the World £135.00 (Airmail)**

To: Practical Electronics
Wimborne Publishing Ltd
113 Lynwood Drive, Wimborne, Dorset BH21 1UU

Tel: 01202 880299 Fax: 01202 843233
Email: stewart.kearn@wimborne.co.uk

I enclose payment of £ ....
(cheque/PO in £ sterling only)
payable to *Practical Electronics*

☐ Please charge my Visa/Mastercard

My card number is: ....
Please print clearly, and check that you have the number correct

Card Security Code .... Valid From Date........................
(The last 3 digits on or just under the signature strip)

Card Ex. Date ....

Name ....

Address ....

Post code .... Tel. ....

Email ....

Subscriptions start with the next available issue.

Accelerate Your Design With Our Comprehensive Ecosystem

We Have What You Need, When You Need It

Looking for the fastest, simplest and lowest-risk path from prototype to production? Microchip offers you complete design support for every stage of your project with our complete development ecosystem.

- Build prototypes quickly with an intuitive design and debugging environment
- Jump-start your project with reference designs and application-specific hardware
- Lower your risk with proven tools and professionally tested software

No matter your needs, we offer you complete design support at every stage of your project.

Start developing now at
www.microchip.com/Ecosystem

MICROCHIP

The Microchip name and logo and the Microchip logo are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries. All other trademarks are the property of their registered owners.
© 2018 Microchip Technology Inc. All rights reserved. DS30010185A. MEC2229Eng11/18

STORE YOUR BACK ISSUES ON CD-ROMS

VOL 1 **BACK ISSUES** – January 1999 to June 1999
Plus bonus material from Nov and Dec 1998
VOL 2 **BACK ISSUES** – July 1999 to December 1999
VOL 3 **BACK ISSUES** – January 2000 to June 2000
VOL 4 **BACK ISSUES** – July 2000 to December 2000
VOL 5 **BACK ISSUES** – January 2001 to June 2001
VOL 6 **BACK ISSUES** – July 2001 to December 2001
VOL 7 **BACK ISSUES** – January 2002 to June 2002
VOL 8 **BACK ISSUES** – July 2002 to December 2002
VOL 9 **BACK ISSUES** – January 2003 to June 2003
VOL 10 **BACK ISSUES** – July 2003 to December 2003
VOL 11 **BACK ISSUES** – January 2004 to June 2004
VOL 12 **BACK ISSUES** – July 2004 to December 2004
VOL 13 **BACK ISSUES** – January 2005 to June 2005
VOL 14 **BACK ISSUES** – July 2005 to December 2005
VOL 15 **BACK ISSUES** – January 2006 to June 2006
VOL 16 **BACK ISSUES** – July 2006 to December 2006
VOL 17 **BACK ISSUES** – January 2007 to June 2007
VOL 18 **BACK ISSUES** – July 2007 to December 2007
VOL 19 **BACK ISSUES** – January 2008 to June 2008
VOL 20 **BACK ISSUES** – July 2008 to December 2008
VOL 21 **BACK ISSUES** – January 2009 to June 2009
VOL 22 **BACK ISSUES** – July 2009 to December 2009

ONLY £16.45 each
Five years just £29.95
including VAT and p&p

VOL 23 **BACK ISSUES** – January 2010 to June 2010
VOL 24 **BACK ISSUES** – July 2010 to December 2010
VOL 25 **BACK ISSUES** – January 2011 to June 2011
VOL 26 **BACK ISSUES** – July 2011 to December 2011
VOL 27 **BACK ISSUES** – January 2012 to June 2012
VOL 28 **BACK ISSUES** – July 2012 to December 2012
VOL 29 **BACK ISSUES** – January 2013 to June 2013
VOL 30 **BACK ISSUES** – July 2013 to December 2013
VOL 31 **BACK ISSUES** – January 2014 to June 2014
VOL 32 **BACK ISSUES** – July 2014 to December 2014
VOL 33 **BACK ISSUES** – January 2015 to June 2015
VOL 34 **BACK ISSUES** – July 2015 to December 2015
VOL 35 **BACK ISSUES** – January 2016 to June 2016
VOL 36 **BACK ISSUES** – July 2016 to December 2016
VOL 37 **BACK ISSUES** – January 2017 to June 2017
VOL 38 **BACK ISSUES** – July 2017 to December 2017
VOL 39 **BACK ISSUES** – January 2018 to June 2018

FIVE-YEAR DVD/CD-ROMs

No.1 – Jan 03 to Dec 07
No.2 – Jan 04 to Dec 08
No.3 – Jan 05 to Dec 09
No.4 – Jan 06 to Dec 10
No.5 – Jan 07 to Dec 11
No.6 – Jan 08 to Dec 12
No.7 – Jan 09 to Dec 13
No.8 – Jan 10 to Dec 14
No.9 – Jan 11 to Dec 15
No.10 – Jan 12 to Dec 16
No.11 – Jan 13 to Dec 17 – NEW!

15 YEARS OF EPE!

AMAZING
BUNDLE OFFER

No.1 – Jan 03 to Dec 07
PLUS
No.6 – Jan 08 to Dec 12
PLUS
No.11 – Jan 13 to Dec 17

ALL FOR JUST
£39.95

These DVD/CD-ROMs are suitable for any PC with a DVD/CD-ROM drive.
They require Adobe Acrobat Reader (free from: www.adobe.com/acrobat)

ORDER YOUR CD-ROMs TODAY!
JUST CALL 01202 880299 – OR VISIT www.epemag.com

Practical Electronics

Volume 48. No. 6
June 2019
ISSN 0262 3617

Editorial

Editorial offices
Practical Electronics
Electron Publishing Limited
1 Buckingham Road
Brighton
East Sussex BN1 3RA

Tel 01273 777619
Mob 07973 518682
Fax 01202 843233
Email pe@electronpublishing.com
Web www.epemag.com

Advertisement offices
Practical Electronics Adverts
1 Buckingham Road
Brighton
East Sussex BN1 3RA

Phone 01273 777619
Mob 07973 518682
Email pe@electronpublishing.com

Editor Matt Pulzer
General Manager Louisa Pulzer
Subscriptions Stewart Kearn
Online Editor Alan Winstanley
Publisher Matt Pulzer

Technical enquiries
We regret technical enquiries cannot be answered over the telephone. We are unable to offer any advice on the use, purchase, repair or modification of commercial equipment or the incorporation or modification of designs published in the magazine. We cannot provide data or answer queries on articles or projects that are more than five years old.

Questions about articles or projects should be sent to the editor by email: pe@electronpublishing.com

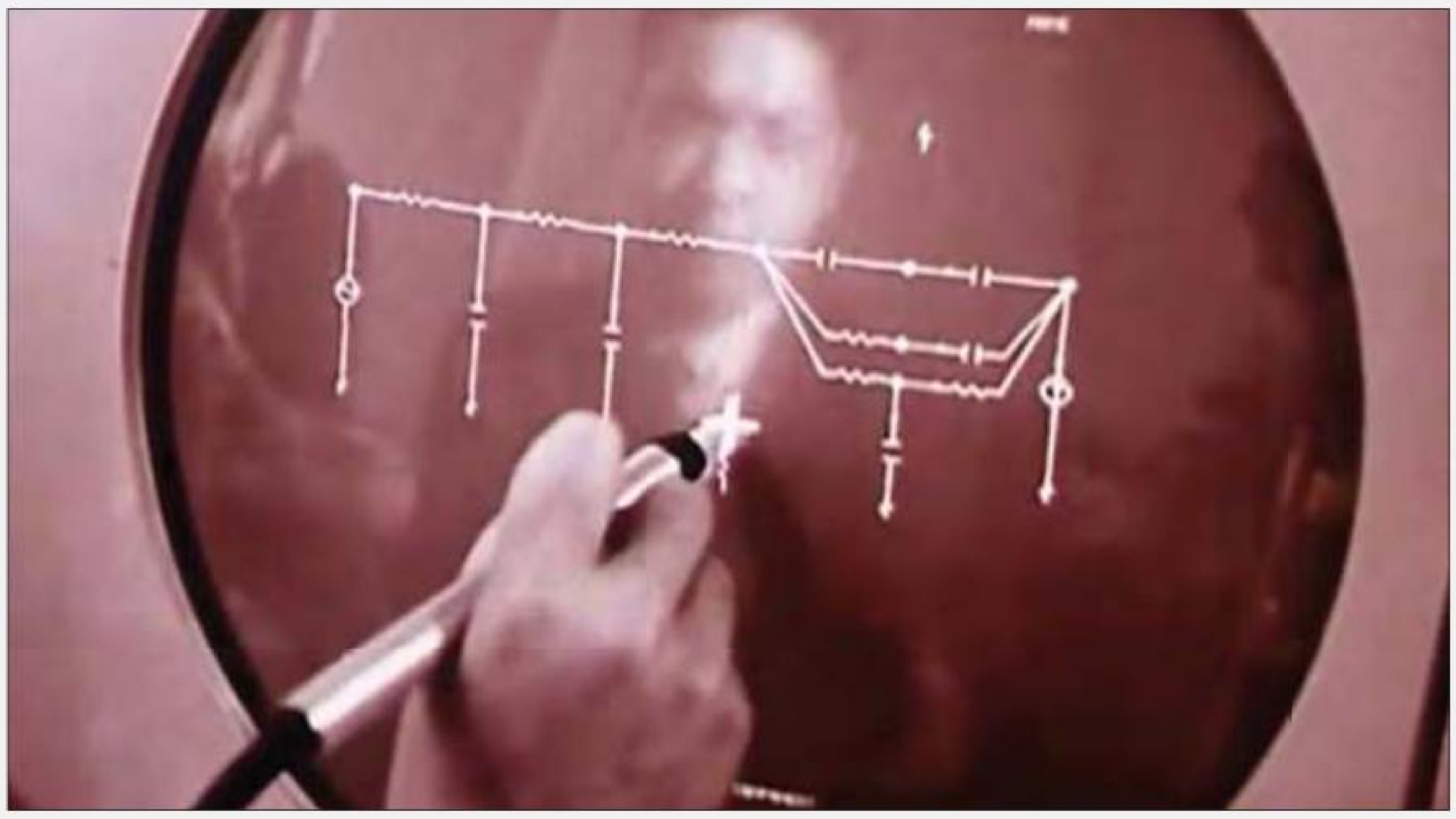
Projects and circuits
All reasonable precautions are taken to ensure that the advice and data given to readers is reliable. We cannot, however, guarantee it and we cannot accept legal responsibility for it.

A number of projects and circuits published in *Practical Electronics* employ voltages that can be lethal. You should not build, test, modify or renovate any item of mains-powered equipment unless you fully understand the safety aspects involved and you use an RCD (GFCI) adaptor.

Component supplies
We do not supply electronic components or kits for building the projects featured, these can be supplied by advertisers. We advise readers to check that all parts are still available before commencing any project in a back-dated issue.

Advertisements
Although the proprietors and staff of *Practical Electronics* take reasonable precautions to protect the interests of readers by ensuring as far as practicable that advertisements are bona fide, the magazine and its publishers cannot give any undertakings in respect of statements or claims made by advertisers, whether these advertisements are printed as part of the magazine, or in inserts. The Publishers regret that under no circumstances will the magazine accept liability for non-receipt of goods ordered, or for late delivery, or for faults in manufacture.

Transmitters/bugs/telephone equipment
We advise readers that certain items of radio transmitting and telephone equipment which may be advertised in our pages cannot be legally used in the UK. Readers should check the law before buying any transmitting or telephone equipment, as a fine, confiscation of equipment and/or imprisonment can result from illegal use or ownership. The laws vary from country to country; readers should check local laws.



50 years of progress

A common criticism of the web is that it is stuffed with time-wasting nonsense and not nearly enough high-quality content. YouTube may be the natural home of some pretty terrible kitten videos (and much worse), but with patience and persistence it is also the go-to place to find fascinating film from the pre-digital era that conscientious libraries have taken the time to digitise and share. I am currently enjoying fascinating short films from the 1950s and 60s explaining the progress of electronic technology in general, and of course computers in particular.

A great example is *The Incredible Machine* (**https://youtu.be/iwVu2BWLZqA**) from Bell Telephone Laboratories, shot in 1968. What I find remarkable is just how relevant and advanced research was in the late 60s. The screen grab above shows a circuit simulation that is not a million miles from the LTspice system Ian Bell regularly discusses in *Circuit Surgery*.

The simulation ran on a Bell Labs Graphic 1 computer system, which consisted of a DEC (Digital Equipment Corporation) PDP-5 computer coupled with input devices such as the Type 370 light pen and Teletype Corporation Teletype Model 33 keyboard, married to a DEC Type 340 precision incremental display. This was backed by a 36-bit Ampex RVQ buffer memory capable of storing 4096 words.

The resolution on the monitor was 1024 × 1024, which seems pretty good for such an early system. This interface was designed to transform the graphics-based input into output to be fed into a IBM 7094 running at 200Kflop/s. The whole thing was attached to a microfilm-based recorder – the Stromberg Carlson SC 4020 – which took hours to read and record the data.

The film also covers work on digital musical composition, digital movie production, 3D simulation of an orbiting satellite, converting pictures to 'mosaics composed of tiny images' and digital voice modulation.

Pretty much every topic covered is now a major industry in its own right – truly, mighty oaks grow from the tiny acorns of such research.

Matt Pulzer
Publisher

NEWS

A roundup of the latest news from the world of electronics and technology

Microchip Snap

Mike Hibbett reports

Those of you who have been following developments at Microchip may have heard of the Microchip Snap – their newest, lowest-cost programmer/debugger. We were fortunate enough to snag one at the Embedded World conference this year.

Unlike their PICkit products it's delivered as a simple PCB (see photo below). Available from all distributors (eg, Farnell) for just £12 plus VAT, it's a great deal. The PICkit 3 and PICkit 4 are already very low cost, but the Snap takes it to a new low-cost standard.

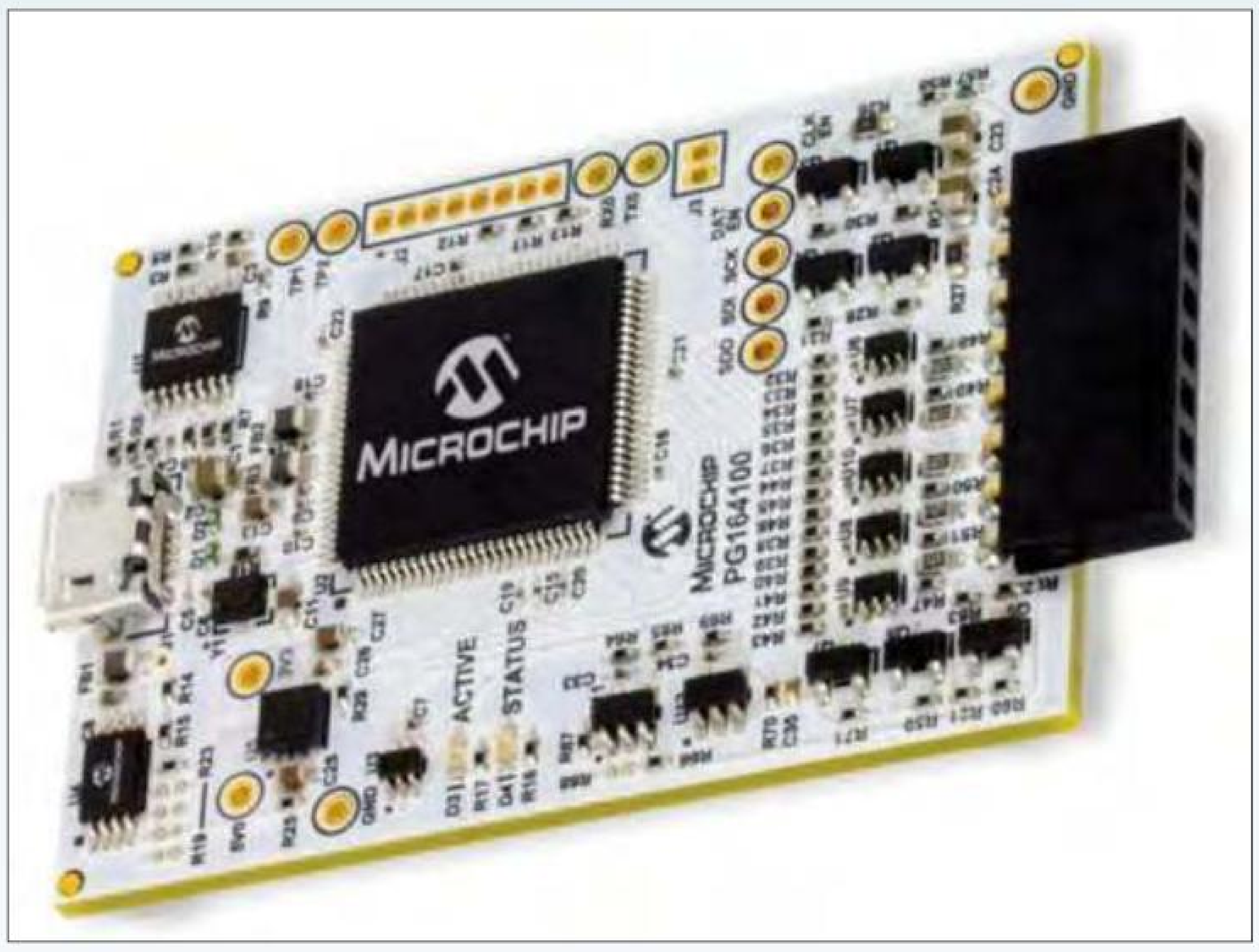
The MPLAB Snap In-Circuit Debugger/Programmer supports most, but not all, PIC, AVR and dsPIC DSCs, and the good news is that the firmware is continually being upgraded to add support for new devices. The bad news is that *PE's* current favourite minimalist PIC, the PIC10F202 is not yet supported by the Snap, but we hope and expect that will change over time.

MPLAB X

The Snap uses the powerful graphical user interface of MPLAB X Integrated Development Environment (IDE) version 5.05 or later. It connects to a computer using a high-speed USB 2.0 interface and to the target via an 8-pin SIL connector. The Snap's connector uses two device I/O pins and the reset line to implement in-circuit debugging and In-Circuit Serial Programming (ICSP). In short, it has all the speed and features entry-level users need to quickly debug their project prototypes.

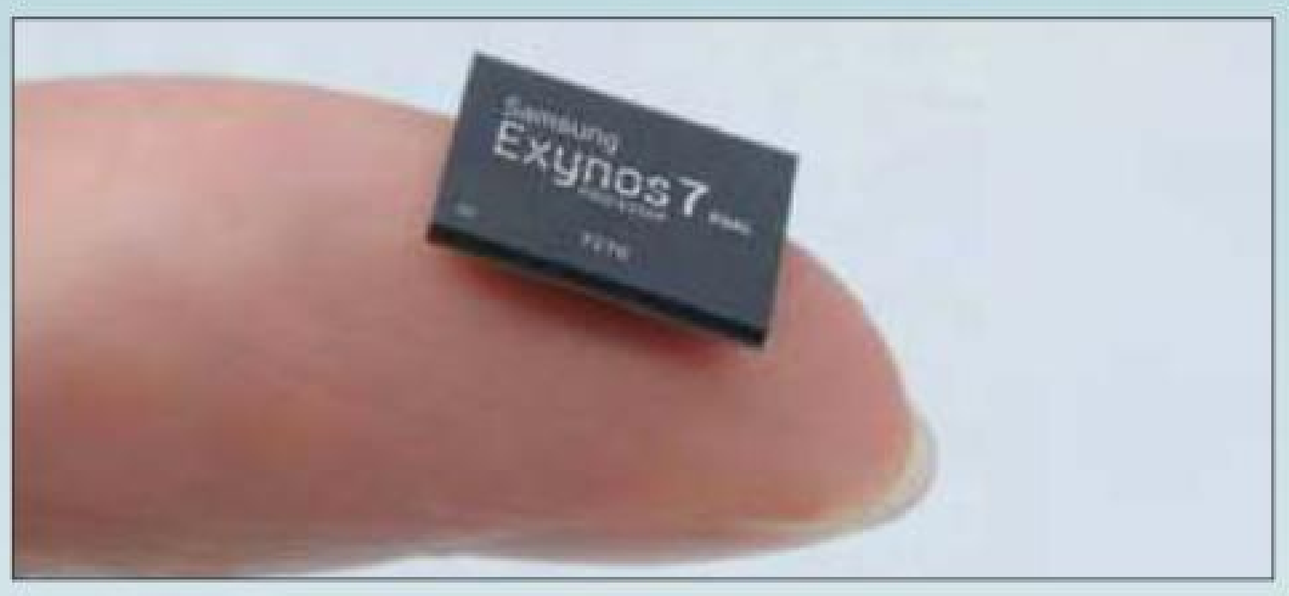
The Snap supports advanced interfaces such as 4-wire JTAG and Serial Wire Debug with streaming data gateway, while also being backward compatible for demo boards, headers and target systems that use the 2-wire JTAG and ICSP.

It looks like a great product and we will discuss and explore the Microchip Snap more fully in future *PIC n' Mix* articles – watch out for more!



Microchip's Snap – their lowest-cost programmer/debugger

Down to 5nm



April 2019 was a good month for aficionados of Moore's Law (the observation that the number of transistors in ICs doubles 'roughly' every two years). Two major chip foundries – TSMC from Taiwan and South Korea's Samsung – announced success with 5nm technology, continuing the sector's decades of increasing miniaturisation, albeit at a slowing pace. Compared to TSMC's 7nm process, the new scale generation promises 1.8 times the logic density, with a 15% speed gain.

TSMC said earlier this year that its 5nm process would be ready for volume production by the end of 2020. Apple's 2020 iPhone is expected to use TSMC's 5nm ICs, produced using EUV (extreme ultraviolet) lithography.

So where do we go from here? There is plenty of impressive research. In 2016, researchers at Berkeley Lab created a transistor with a working 1nm gate (the device's drain was considerably larger). In early 2018, IMEC reported 3nm test chips, and Samsung have announced plans to produce 3nm FETs in 2021. However, the king of miniature transistors currently resides at the Karlsruhe Institute of Technology, which in 2018 created a transistor with a working single-atom gate.

HAMMOND MANUFACTURING®

Plastic enclosures standard & miniature

www.hammondmfg.com/dwg8.htm
www.hammondmfg.com/1551USB.htm

01256 812812
sales@hammond-electronics.co.uk

Colossus at 75

The 75th anniversary of the first attack by the Colossus computer on a Second World War enemy Lorenz message has coincided with the discovery of a selection of original decrypts showing the nature of the intelligence that was intercepted. The decrypts are now being analysed by researchers for a special display in the Colossus Gallery at The National Museum of Computing (TNMOC) in May to mark the 75th anniversary of the arrival of Colossus Mk II in time for D-Day in 1944.

On 5 February 1944, Colossus Mk I attacked its first message encrypted by Lorenz, the highly sophisticated cipher used in communications between Hitler and his generals during the Second World War.

Lifesaver

Designed by brilliant British telecommunications design engineer Tommy Flowers, Colossus was built to speed up code-breaking of the hugely complex Lorenz cipher. By the end of the war there were ten functioning Colossi and they made a decisive impact in shortening the war and saving countless lives.

The first electronic computer

News of the existence of the Colossus, widely regarded as the first electronic computer, was kept top secret for 30 years partly because of the sophistication of its methods to help break Lorenz messages by finding the frequently changing wheel patterns of the Lorenz encryption machine.

Standing 2m high, 5m wide and almost 4m deep, Colossus weighed five tonnes, and used 8kW of power. Colossus Mark II, which arrived at Bletchley Park in May 1944, had 2,500 valves and 7km of wiring, and was able to read 5000 characters per second (faster than anything that had been produced commercially).

By accelerating the discovery of the wheel patterns of the ever-changing Lorenz-encrypted messages, 63 million characters of high-grade German messages had been decrypted by 550 people working on the ten functioning Colossus machines at Bletchley Park by the end of the war.

The full impact of Colossus continues to be revealed. Although its existence was kept secret until 1975, many of those who worked on Colossus went on to build other computers and technologies, confident in the knowledge that it was possible to build reliable machines of such complexity using valves.

Andrew Herbert, TNMOC's chair said, 'The working rebuild of Colossus at The National Museum of Computing is an inspiration to all our visitors who are astonished when they see the working machine and learn of its impact.

"The achievements of those who worked at wartime Bletchley Park are humbling. Despite decades of secrecy, the names of the key characters involved with the breaking of Lorenz and the construction of Colossus are becoming increasingly well known –Bill Tutte who deduced how the Lorenz machine worked, Colonel Tester and the Testery who broke the cipher by hand, and Tommy Flowers who designed Colossus to speed up the code-breaking process.'

XPired

Microsoft Windows XP has finally expired, nearly 18 years after it was first launched in 2001. The popular and very-long-lived operating system's final publicly supported variant – Windows Embedded POSReady 2009 – reached the end of its life cycle support on 9 April 2019. RIP XP.

Fluke/Philips PM3092 Oscilloscope
2+2 Channel 200MHz Delay TB,
Autoset etc – £250

STEWART OF READING
17A King Street, Mortimer, near Reading, RG7 3RS
Telephone: **0118 933 1111** Fax: **0118 933 2375**
USED ELECTRONIC TEST EQUIPMENT
Check website **www.stewart-of-reading.co.uk**
(ALL PRICES PLUS CARRIAGE & VAT)
Please check availability before ordering or calling in

HP 54600B Oscilloscope
Analogue/Digital Dual Trace 100MHz
Only £75, with accessories £125

LAMBDA GENESYS	PSU GEN100-15 100V 15A Boxed As New	£400
LAMBDA GENESYS	PSU GEN50-30 50V 30A	£400
IFR 2025	Signal Generator 9kHz – 2.51GHz Opt 04/11	£900
IFR 2948B	Communication Service Monitor Opts 03/25 Avionics	POA
IFR 6843	Microwave Systems Analyser 10MHz – 20GHz	POA
R&S APN62	Syn Function Generator 1Hz – 260kHz	£295
Agilent 8712ET	RF Network Analyser 300kHz – 1300MHz	POA
HP8903A/B	Audio Analyser	£750 – £950
HP8757D	Scaler Network Analyser	POA
HP3325A	Synthesised Function Generator	£195
HP3561A	Dynamic Signal Analyser	£650
HP6032A	PSU 0-60V 0-50A 1000W	£750
HP6622A	PSU 0-20V 4A Twice or 0-50V 2A Twice	£350
HP6624A	PSU 4 Outputs	£400
HP6632B	PSU 0-20V 0-5A	£195
HP6644A	PSU 0-60V 3.5A	£400
HP6654A	PSU 0-60V 0-9A	£500
HP8341A	Synthesised Sweep Generator 10MHz – 20GHz	£2,000
HP83630A	Synthesised Sweeper 10MHz – 26.5 GHz	POA
HP83624A	Synthesised Sweeper 2 – 20GHz	POA
HP8484A	Power Sensor 0.01-18GHz 3nW-10µW	£75
HP8560E	Spectrum Analyser Synthesised 30Hz – 2.9GHz	£1,750
HP8563A	Spectrum Analyser Synthesised 9kHz – 22GHz	£2,250
HP8566B	Spectrum Analsyer 100Hz – 22GHz	£1,200
HP8662A	RF Generator 10kHz – 1280MHz	£750
Marconi 2022E	Synthesised AM/FM Signal Generator 10kHz – 1.01GHz	£325
Marconi 2024	Synthesised Signal Generator 9kHz – 2.4GHz	£800
Marconi 2030	Synthesised Signal Generator 10kHz – 1.35GHz	£750
Marconi 2023A	Signal Generator 9kHz – 1.2GHz	£700

Marconi 2305	Modulation Meter	£250
Marconi 2440	Counter 20GHz	£295
Marconi 2945/A/B	Communications Test Set Various Options	POA
Marconi 2955	Radio Communications Test Set	£595
Marconi 2955A	Radio Communications Test Set	£725
Marconi 2955B	Radio Communications Test Set	£800
Marconi 6200	Microwave Test Set	£1,500
Marconi 6200A	Microwave Test Set 10MHz – 20GHz	£1,950
Marconi 6200B	Microwave Test Set	£2,300
Marconi 6960B	Power Meter with 6910 sensor	£295
Tektronix TDS3052B	Oscilloscope 500MHz 2.5GS/s	£1,250
Tektronix TDS3032	Oscilloscope 300MHz 2.5GS/s	£995
Tektronix TDS3012	Oscilloscope 2 Channel 100MHz 1.25GS/s	£450
Tektronix 2430A	Oscilloscope Dual Trace 150MHz 100MS/s	£350
Tektronix 2465B	Oscilloscope 4 Channel 400MHz	£600
Farnell AP60/50	PSU 0-60V 0-50A 1kW Switch Mode	£300
Farnell XA35/2T	PSU 0-35V 0-2A Twice Digital	£75
Farnell AP100-90	Power Supply 100V 90A	£900
Farnell LF1	Sine/Sq Oscillator 10Hz – 1MHz	£45
Racal 1991	Counter/Timer 160MHz 9 Digit	£150
Racal 2101	Counter 20GHz LED	£295
Racal 9300	True RMS Millivoltmeter 5Hz – 20MHz etc	£45
Racal 9300B	As 9300	£75
Solartron 7150/PLUS	6½ Digit DMM True RMS IEEE	£65/£75
Solatron 1253	Gain Phase Analyser 1mHz – 20kHz	£600
Solartron SI 1255	HF Frequency Response Analyser	POA
Tasakago TM035-2	PSU 0-35V 0-2A 2 Meters	£30
Thurlby PL320QMD	PSU 0-30V 0-2A Twice	£160 – £200
Thurlby TG210	Function Generator 0.002-2MHz TTL etc Kenwood Badged	£65

HP/Agilent HP 34401A Digital
Multimeter 6½ Digit £325 – £375

HP33120A	Function Generator 100 microHz – 15MHz	£350
HP53131A	Universal Counter 3GHz Boxed unused	£600
HP53131A	Universal Counter 225MHz	£350
Audio Precision	SYS2712 Audio Analyser – in original box	POA
Datron 4708	Autocal Multifunction Standard	POA
Druck DPI 515	Pressure Calibrator/Controller	£400
Datron 1081	Autocal Standards Multimeter	POA
ENI 325LA	RF Power Amplifier 250kHz – 150MHz 25W 50dB	POA
Keithley 228	Voltage/Current Source	POA
Time 9818	DC Current & Voltage Calibrator	POA

Marconi 2955B Radio
Communications Test Set – £800

New Light on LEDs

Techno Talk

Mark Nelson

Have you converted to LED lamp bulbs yet? Prices are tumbling and the economics emphatically confirm replacing your incandescent lamps is an absolute no-brainer. The Victorians installed gas lamps to deal with objectionable smells; today they are getting the LED treatment. If that sounds ridiculous, keep reading! And there's a dramatic development in blocking junk phone calls… only not in the UK.

One of my favourite channels on YouTube is hosted by Big Clive, who delivers destructive teardowns of electronic gadgets with both humour and wisdom. If you haven't discovered his excellent videos yet, I implore you to sample an amazing 1,422 videos in what he himself calls 'the trashiest electronic channel on YouTube, see: **http://bit.ly/pe-jun19-tt1**

Clive of YouTube

One of the insights I gained from Clive is that the cheaper LED lamps sold at places like Poundland don't have chopper power supplies, but rely on capacitive droppers instead, eliminating nasty switch-mode interference! We should all be in favour of getting rid of switch-mode power supplies, as this makes the lamps cheaper to buy, cooler, cheaper to run and much more like conventional incandescent bulbs (and thus the right size and delivering all-round light distribution). Why on earth settle for a hot 9W bulb when you can have a cool 7W one instead? That said, even a cool LED lamp will 'enjoy' a relatively short life when installed in enclosed fittings such as some porch lights, where they slowly 'cook'.

You do the math

Contrary to received wisdom, don't expect LED bulbs to last significantly longer than their incandescent equivalents. However, now that they're as cheap as chips, it still makes sense to switch to LED bulbs, even if they fail after only 1,000 hours. Really? Yes, just do the maths:

- 60W incandescent bulb run for 1,000 hours @ 15p/kWh = £9.00. Now add the £1 cost of the bulb, giving total life cost of £10.
- 7W 806-lumen filament LED run for 1,000 hours @ 15p/kWh = £1.05. Now add the £5 cost of the bulb, giving total life cost of £6.05.

Although you cannot find truly low-cost LED lamps in every supermarket, Screwfix sells many LED bulbs at £2.49 (other sources are available). My thanks go to loyal readers Gerry and Chris for this enlightenment.

If you were an early adopter of LED lamps (as I was), you may have bad memories of examples that went ping and abruptly died the second time you switched them on. It's true; they were not very reliable, but you did get replacements if you took them back with the sales receipt. Now, however, they are far more robust, as Allan G3PIY expounded on a military radio forum recently. He reports: 'I was given some failed LED mains lamps recently. The chopper power supplies had failed. The LED chips were mounted on pieces of aluminium with welded wires and I decided to see how robust they were. Connecting a low voltage variable DC power supply, I set the voltage and the LEDs came on pretty bright. I cranked up the voltage and they got extremely bright… then a bit more and they drew over an amp and were too bright to look at. Then the LEDs unsoldered themselves from the aluminium and the welding to the leads melted and they went off. But the actual semiconductors hadn't failed, even though they left burn marks on the bench. Very impressive!'

Nasty niffs

The sweltering summer of 1858 marked the arrival of the 'Great Stink of London', in which the stench of human waste floating down the polluted River Thames was so offensive that Members of Parliament were forced to quit the House of Commons. Back then, sewage poured into the river and was carried back and forth by the tides. Parliament was forced to act to improve London's primitive system of sanitation. Many people were convinced that foetid odours produced disease ('malaria' is a contracted form of the Italian *mala aria,* or 'bad air'). One remedy was the invention of the 'sewer gas destructor lamp'. Looking like a normal gas street lamp, this invention burnt normal town gas in a flame that drew up noxious gases from the sewers below and discharged them well above the heads of passers-by to dissipate foul odours.

Today these devices are rare curiosities and are now conserved and maintained by local authorities (see: **http://bit.ly/pe-jun19-tt2** for locations). Since it appears that we produce fewer 'noxious odours' today, an innovative company, Pudsey Diamond Engineering Ltd, is converting the lamps to solar-powered operation, using LEDs that deliver a colour of light closely matching that produced by a gas burner. The solar panels are fitted unobtrusively, with batteries and electronics hidden beneath them. The system is completely self-contained, requires no ongoing maintenance and is designed with sufficient spare battery capacity to maintain operation through the winter months, when there is less solar energy available. More at: **http://bit.ly/pe-jun19-tt3** – leaving only one question: how do we eliminate residual nasty niffs now?

The US can do it, why can't we?

Following my February rant about OFCOM's reluctance to force telephone companies to block fraudulent telephone calls made from non-existent numbers, it's interesting to contrast this inaction with the reaction in the US. Over there, their telephone networks now face sanctions if they don't prevent caller ID spoofing. The FCC (Federal Communications Commission – the US watchdog body) has demanded that phone companies implement a robust caller ID authentication system this year to combat illegal spoofing – or face legal enforcement. Says FCC chairman Ajit Pai, 'American consumers are sick and tired of unwanted robocalls. Caller ID authentication will be a significant step towards ending the scourge of spoofed robocalls. It's time for carriers to implement robust caller ID authentication. This goal should be achievable for every major wireless provider, interconnected VoIP operator and telephone company – and if imajor carriers won't meet the deadline to get this done this year, the FCC will have to consider regulatory intervention.'

Well done, that man. How about similar action on this side of the Atlantic?

pico Technology

TC-08: Your 8-Channel Thermocouple Data Logger

1 Connect

2 Configure

3 Collect

Take your first temperature logger measurements in three simple steps

TC-08
23.27 °C
pico Technology

When paired with the power and astounding flexibility of PicoLog 6 software, the TC-08 temperature logger can start recording reliable, accurate, detailed measurements in under 60 seconds.

Only £289 (excl VAT)

To find out more visit: **www.picotech.com/A315**

Net Work

Alan Winstanley

This month, *Net Work* does exactly what it says on the tin. Alan Winstanley examines options for updating, extending and (hopefully) simplifying home-networked Internet access.

Last month, I mentioned the 'sunk cost fallacy'; the age-old dilemma that many of us face when deciding whether to stick with old ideas and solutions or scrap everything and start over again. What if your network system contains years of still-functioning hardware? Should you cut losses and update? Or do you carry on, potentially throwing good money after bad?

Like many readers, the author's network contains a host of wired and wireless devices that have multiplied over the years and the maxim, 'if it ain't broke, don't fix it' generally rules. This network includes smart TVs, PVRs, Internet radios, smartphones, smart TV dongles, smart lights and sockets, tablets, NAS drives, devolo Powerline adaptors, IP cameras, Wi-Fi printers, laptops and an array of desktop PCs. The list goes on, and other users also increasingly rely on cloud-powered smart devices to control their central heating, video doorbell or security system.

Just like some plumbers seem to add more pipework rather than streamline or modernise the plumbing, it's usually easier and cheaper just to keep hooking more devices onto an existing network and steer clear of modernising the core network altogether. It's simple to add a gigabit switch (yet more wires and another mains adaptor!) to a router to give a burgeoning network more Ethernet ports. Adding more devices frequently creates all sorts of time-consuming problems that grab our attention, but smart devices are thankfully becoming 'smarter' to install, with setup being handled wirelessly thanks to a user-friendly app running on a smartphone.

From network to not work

Sometimes it takes a persistent problem to force a wider review of network hardware. One of the writer's desktop PCs suffers from regular network drop-outs, making web surfing frustrating at times. The computer depends on wireless connectivity, but Wi-Fi coverage isn't exactly helped by the copper heating system tank situated next door, coupled to lots of plumber's pipework that resembles something from the engine room of *The African Queen*! Distance, outside interference and general traffic congestion can also degrade the Wi-Fi signal. In this example it's not possible to run an Ethernet cable from the router, but a Wi-Fi repeater could be installed somewhere, or the LAN could run through a HomePlug adaptor (Ethernet through the ring main) instead, assuming a clean raw mains connection is available near the client PC.

How Develo can you go

In fact, the home network is already equipped with a devolo dLAN 500 plug-in adaptor that 'feeds' Internet access into the ring mains, with a separate adaptor providing wired Ethernet ports for a smart TV and PVR and also offering a wireless access point in the lounge. These plug-in 'pass through' HomePlug adaptors also carry a handy mains outlet on the front. A simple Windows program (devolo Cockpit) is used for basic set-up and monitoring, and the Powerline system has proven extremely reliable over several years. However, due to the scathing reviews the 'My devolo' Android app has received from users, the author decided not to try it on a smartphone.

The 'gotcha' is that, like all such technology, the boilerplate '500' megabits per second is the maximum *theoretical* rate achievable, but the devolo software reports a throughput that's just a fraction of that, typically 60 Mbps or so over some 40-year-old mains wiring. Even so, there have been no issues when watching (eg, streaming HD programs or movies on demand when using the devolo dLAN 500 HomePlug system on a smart TV) but more demanding users might need something faster.

A low-price devolo Wi-Fi adaptor (£25/$33 each) could act as a quick and easy add-on wireless extender for the troublesome PC, and it provides a 100Mb LAN port as well. In my case it could plug directly into an unused mains socket in the same room and seems like a good workaround. Other makes of HomePlug-type devices are available, but it's probably best to stick to one brand throughout. Powering them on mains extension leads or using 'noisy' loads like fans or motors can degrade performance and is best avoided. Simple Ethernet-only (no Wi-Fi) Powerline adaptors are also available (£20/$25 each) and money-saving multipacks are on sale. The HomePlug approach seems to be the simplest plug-and-play way of extending Internet connectivity wirelessly provided the mains is on the same electrical phase throughout and is of a 'clean' noise-free quality. However, it may not perform well over ancient wiring.

AC rules the waves

Most home Internet users operate their network in the classic way: a modem provides an Ethernet port from the incoming broadband connection, and a wireless router networks client devices together behind a firewall. It's very satisfying when peripherals such as routers turn out to be future proof just when you need it the most: the writer's eight-year-old Billion gigabit router is a robust dual WAN type which used ADSL2+ until fibre broadband finally took over. The router handled the changeover seamlessly in a rare and gratifying example of user-friendliness. Usually, a modem and router are combined into one unit (though not in the writer's case) which also provides 2.4GHz Wi-Fi. An older dual-band 802.11n router offers both 2.4GHz and less crowded 5GHz Wi-Fi, while a modern tri-band router

devolo
500Mbps

devolo Powerline distributes Internet access via the ring main. This UK adaptor offers Ethernet ports and Wi-Fi access.

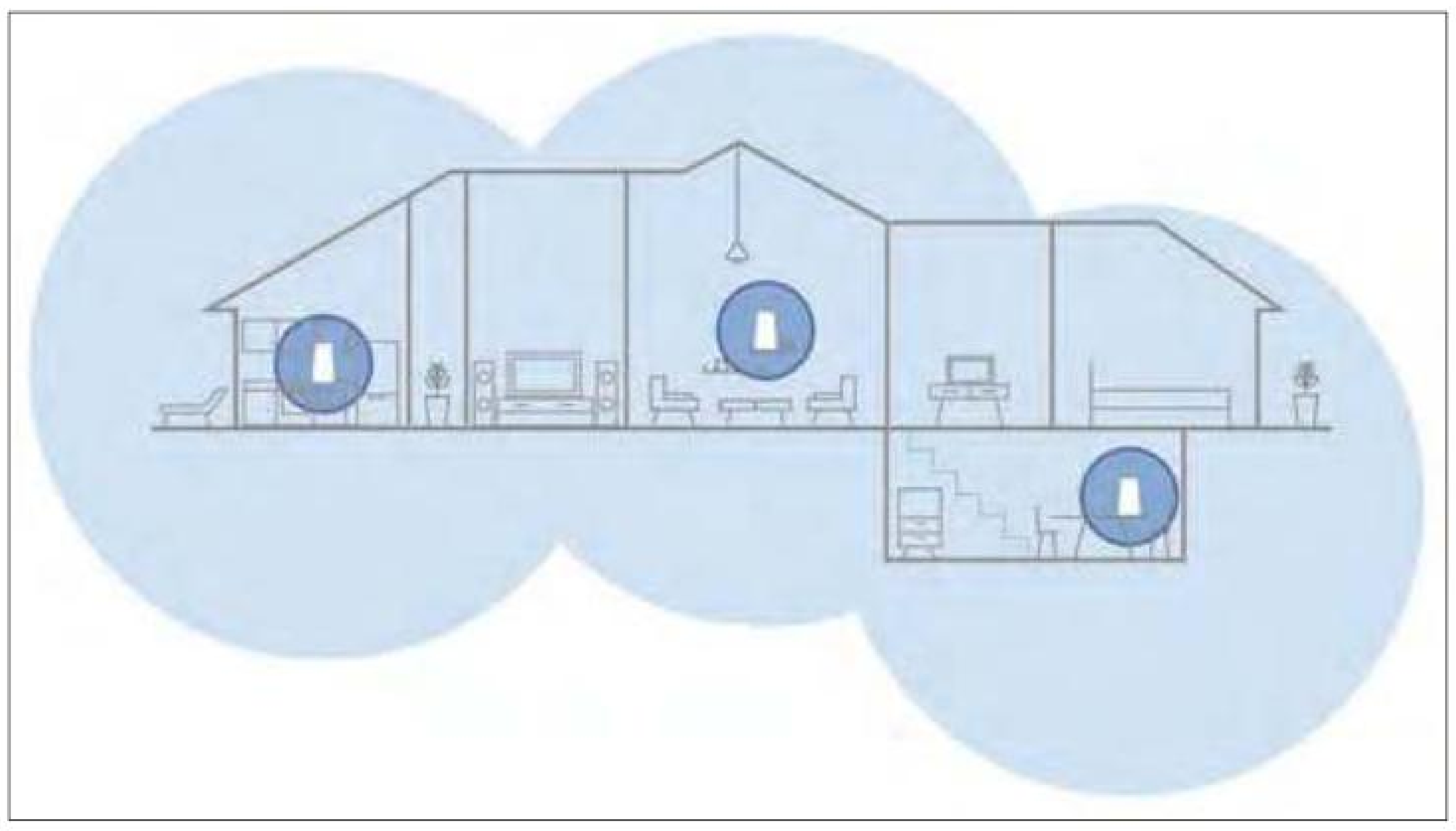
has 2.4GHz and two 5GHz radios to cater for more demanding needs such as gaming, VR or video streaming. Of course, client devices themselves need to be 5GHz-compatible as well.

Ideally, a wireless router should be located centrally for maximum coverage. Stuffing it under a desk, by a TV or at the far edge of a building obviously affects Wi-Fi performance due to the distance and obstacles like pipework, cables, thick stonework or foil-lined cavity walls, all of which impede the wireless signal. Hence the plethora of plug-in wireless boosters and repeaters that are supposed to increase the reach of Wi-Fi, but setting them up creates complications of their own and a repeater cuts the speed in half. Furthermore, using multiple SSIDs (service set identifier) can make life difficult for users when they move around a property.

Although a network is only as fast as its slowest part, for those considering updating a legacy 802.11n router, an 802.11ac spec heralds a much faster type of 5GHz router. A dual-band type is backwards-compatible with legacy 2.4GHz devices and in terms of speed an 802.11ac router is likely to be future proof for years to come. A further clue about a router's performance might be found in the technical nomenclature, but don't be misled: look for 'AC3200' or 'AC5400' for example, but a higher number does not mean one 'ac' router is faster than another; this number refers to the theoretical combined bandwidth that the router's radios can handle – ie, the notional total volume of traffic in Mbps rather than its top speed. A very good explanation of this can be found at: **www.bit.ly/pe-jun19-nw1**

Meshed gear

Another wireless solution now gaining traction is a 'mesh network' based on 802.11s, especially if building a home network from scratch. Mesh hardware has been available for a number of years and a range of lower-cost, consumer-friendly products is now on sale, often marketed as 'Whole Home' products. Instead of using a Wi-Fi router the traditional way – which needs Wi-Fi repeaters and multiple SSIDs dotted around to overcome pesky dead zones – a mesh network uses a clutch of devices (or nodes) to provide more of a blanket wireless coverage around a property. The nodes still need siting properly but, in effect, users just see one network throughout the home. The wireless mesh nodes interact to select the 'best' route for traffic to ensure that users always receive the strongest signal. Built-in Ethernet ports also provide a hard-wired Ethernet access for devices that need it.



Whole Home mesh Wi-Fi promises total coverage around the home (Image: Linksys)

A trio of Linksys Velop triband mesh routers. Ethernet ports are included

All the major comms brands now offer wireless mesh devices, but it's worth knowing that these plug-and-play systems are designed to be user-friendly and some of them may lack some advanced networking features that power users like to see. Starter packs are sold in pairs, a router and a 'satellite' and more satellites can be added (British Telecom (BT) calls theirs, 'discs'). Each Whole Home node typically costs £90-£120 ($120-$150) and brands to consider include the tri-band Linksys Velop, TP-Link Deco, Samsung Connect Home and BT's own Whole Home mesh system with its enigmatic discs. Netgear offers Orbi, which is Alexa- and Google Assistant-aware and has USB and gigabit Ethernet ports. It's important to note that devolo's new Magic Mesh range offers a wireless mesh via Powerline, but sadly it does *not* support older dLAN products like the ones described earlier: so much for future proofing. Some devices are also IFTTT (If This, Then That) compatible, allowing simple routines to be triggered 'smartly', while some include parental controls, device prioritisation, adult website blocking and guest networks. They all seem to rely on an app to help with setup.

Google retails its own dual-band mesh called 'Google Wifi' (a Google account and Android or iOS app are needed). Coverage is a major selling point and Google Wifi suggests one node for a smaller residence (85m^2), two for a medium one (170m^2) and three for up to 420 m^2 to simplify setup. Google and Linksys claim their nodes automatically update for the latest security, which will re-assure some users. Amazon is muscling in on mesh Wi-Fi too, having recently bought the firm Eero, the daddy of low-cost mesh networking. Amazon is expected to integrate Eero wireless mesh technology into its Echo smart speakers, so expect a new product line-up by the year-end.

Not yet a mature technology

The Whole Home mesh market is still evolving and no amount of development is a substitute for launching a product onto millions of users and seeing what happens. Judging by user feedback, some products appear seductively simple to set up but they have their share of niggles and occasionally some serious customer gripes emerge. Some 'gotchas' include complaints of losing Internet access, which means the associated smartphone app couldn't work either. Expert users might also hit thorny problems regarding UPnP, DLNA, VPNs, port forwarding and the problem of 'Double NAT' when hooking up to an existing router (for example, see: **http://bit.ly/pe-jun19-nw2**), so some advance research would be wise prior to investing. The Linksys Velop manual does clearly explain port forwarding, DHCP and MAC filtering too and appears to be a good intermediate-level mesh solution but it has not been tried by the author. (However, the editor tells me he does have the Velop system; it runs well and was, 'no more frustrating to set up than plenty of other systems'!)

Wireless mesh technology has plenty to offer in terms of building a fast, convenient and far-reaching Wi-Fi network, but I suspect many users will keep adding on to their existing network for the time being. If you're starting from scratch or thinking of giving your home network a worthwhile update then Whole Home mesh network is definitely worth investigating.

See you next month for more technology-based reports in *Net Work*!

The author can be reached at:
alan@epemag.net

- **Easy to build/low cost**
- **Arduino based**
- **Ultra-low drift**
- **Auto-drift compensation**
- **Auto L or C identification**
- **Capacitance: 0.1pF to >1F**
- **Inductance: 10nH to >1H**

L:95.63uH 372051Hz
C:0.000 F 707590Hz
--> Inductor
L=95.559uH 80 samp

Wide-range Arduino-based LC Meter

By Tim Blythman

Not only is this new digital *Inductance-Capacitance Meter* easy to build (it's based on a custom Arduino shield and a standard 4-line alphanumeric LCD display), it features very low drift due to a constant self-calibration procedure. Best of all, it has an extended measurement range from less than 1pF to over 1F (yes, ONE FARAD!) for capacitors, and under 100nH to several henries for inductors. You simply must add this to your test equipment arsenal!

A wide-range LC Meter is a very worthwhile device to have on your workbench. But have you tried to buy a good one lately? (Hint – mortgage the kids first!).

Many DMMs have a capacitance meter built in but their range is usually (very) limited. And most cannot measure inductance at all.

Now though, you can have an ultra-wide-range LC meter which has all the 'most wanted' features – high accuracy, very low drift... and best of all, it won't cost you a lot to build. That's mainly because it is based on an Arduino processor. You can choose to build it in a case as a genuine piece of test gear, or assemble it with your Arduino Uno whenever you need to measure a capacitor or inductor.

This design is very accurate because it automatically compensates for its own thermal drift and can measure very small and very large capacitance and inductance values. It automatically senses the component type, so you can connect virtually any capacitor or inductor, big or small, and it will quickly tell you its value on the LCD screen. You can even measure supercapacitors!

This is actually the second *LC Meter* that we have published in the last twelve months. This latest iteration is a big improvement, both in performance and ease-of-use, while being only slightly more complex.

Most importantly, it solves the drift problem that has plagued most DIY *LC Meter* designs – and many commercial designs as well.

Why is measurement drift a problem?

This type of *LC Meter* design has a long history. Our projects in March 2010, and June 2018 were all based on an earlier design by Neil Heckt, which used a PIC16C622 microcontroller. These are all based on an oscillator which incorporates the unknown device (inductor or capacitor) to be measured.

The parameters of the device to be measured (eg, inductance or capacitance) affect the oscillator frequency, and by measuring the change in frequency we can estimate its inductance or capacitance.

This approach first measures the initial oscillator frequency, and then the frequency with the unknown device in circuit. Then the frequency difference is used in a formula to calculate the device's unknown inductance or capacitance value.

The problem is that all those previous designs only measure the initial (default) oscillator frequency when the unit is first powered up. That's because the user has to manually disconnect any components from the test terminals. Unfortunately, as the unit warms up, the oscillator frequency shifts.

So unless you disconnect the device under test (DUT) and 'reboot' the *LC Meter* every time you want to make a new measurement, it won't necessarily be accurate.

That inevitable drift in oscillator frequency means that regardless of how precise the initial calibration may be, each successive measurement is likely to be progressively less accurate.

Our solution is simple: get the microcontroller to frequently disconnect and reconnect the DUT from the circuit. This means it can measure the oscillator frequency with and without the DUT at very short time intervals and compute the difference on this basis. The effect of this is that the *LC*

Features and specifications

Advanced calibration
Continuous drift compensation
Long-term averaging
Inductance range: 10nH to 1H+
Capacitance range: 0.1pF to 1F+ (minimum rating 5V)
Measurement resolution: four significant figures
Component detection: automatic
Sampling rate: Once every two seconds (approx)
Accuracy (when calibrated): within ±1% of reading
Supply voltage: 5-12V DC @ <100mA
Easy-to-assemble, low-cost acrylic case available.

If you're a typical hobbyist you've probably got a pile of capacitors in your junk box and don't know if they're good, bad or indifferent. And even your DMM can't tell you because it won't go high enough, especially for electrolytics. Build this *LC Meter* and you can check them all (even supercaps) – plus all those inductors with no markings!

Meter is constantly compensating for any drift due to temperature, ageing or other factors.

The microcontroller uses reed relays to switch the DUT in and out of circuit. You don't have to do anything to select or control this process; the micro does it automatically.

At the same time, the micro can decide to add some extra components which allow it to make measurements using a different method that's more suitable for higher values of capacitance and inductance.

This is how we've greatly extended its measurement range.

If you want more details on how the oscillator is used to measure the inductor or capacitor value, see Jim Rowe's detailed description in the June 2018 project: *Arduino-based Digital Inductance/Capacitance Meter*.

Extending the measurement range

Our previous designs, and indeed any *LC Meter* based on this circuit configuration, are limited to handling a maximum capacitance value of slightly more than 1µF and a maximum inductance of around 100mH. Values higher than this tend to prevent the oscillator from functioning.

Since there is no easy way to fix the oscillator to solve this range problem, we've used a different method to measure large component values, based on measuring the time constant of an *RC* or *RL* circuit. (A separate panel in this article explains how that method works.)

To allow for this extra measurement mode, we need a way to disconnect the DUT from the oscillator, but luckily, earlier designs had already added that capability using reed relays for the drift cancellation feature.

We also need a way to disconnect the time constant measurement circuitry from the DUT so that it doesn't affect the oscillator – but that turned out to be easy.

The pins on the micro which perform the measurement function are simply set to a high-impedance state when not being used, and they then have virtually no effect on the behaviour of the oscillator.

In the end, the only changes required to add the new measurement mode were two resistors and some extra software routines.

Circuit description

The full circuit diagram is shown in Fig.1. Everything except the Arduino and 20×4 alphanumeric LCD is mounted on the shield board for the Arduino. Let's look first at how the oscillator-based measurements are made.

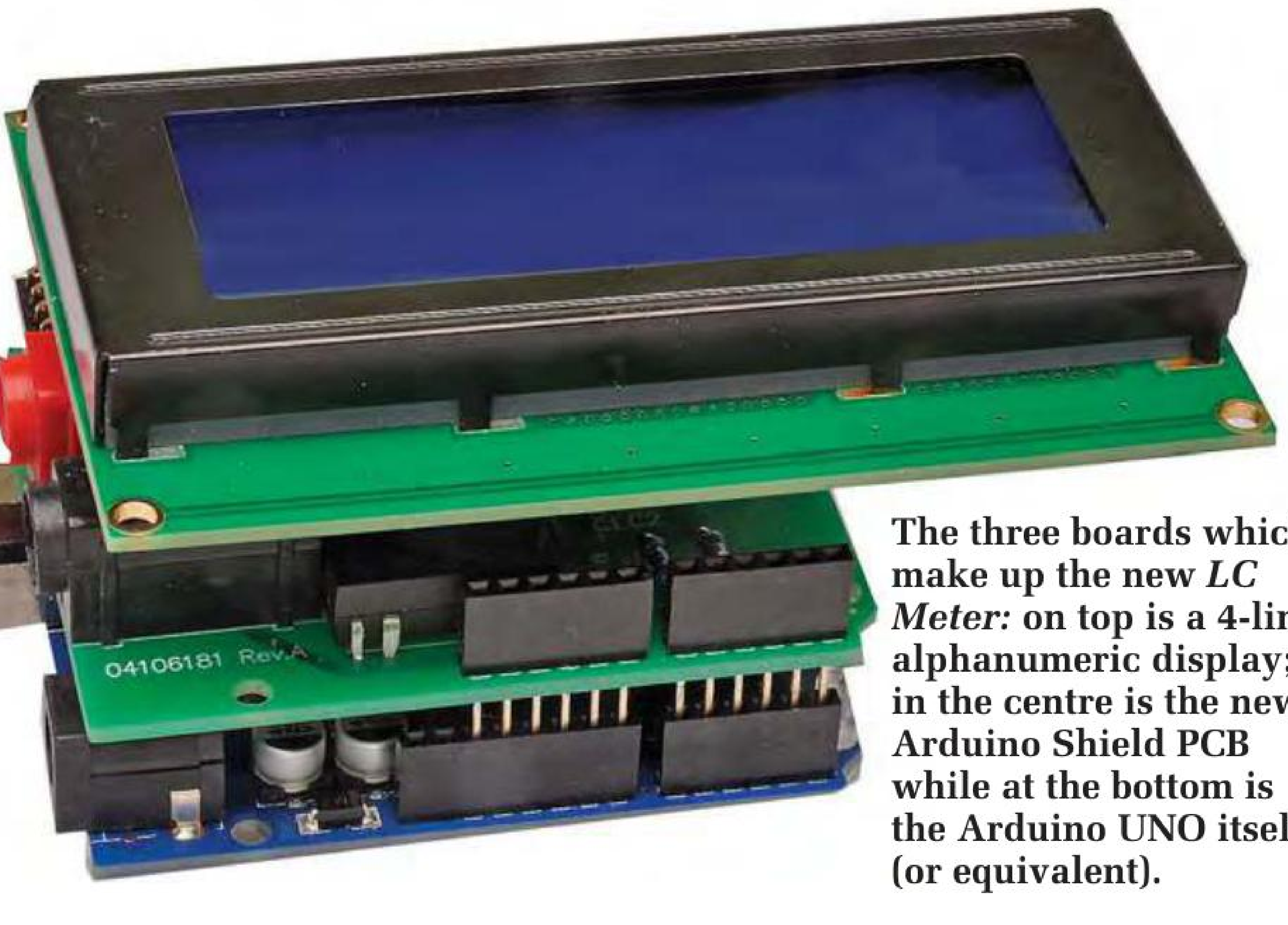
IC1 is a high-speed LM311 comparator, used to drive the resonant circuit into oscillation. The resonant circuit consists of inductor L1 (100µH) and capacitor C1 (1nF). The junction of these two components is coupled to the non-inverting input of IC1, pin 2, via a 10µF capacitor.

Positive feedback is provided around IC1 by a 100kΩ resistor from its pin 7 output to the pin 2 non-inverting input. Pin 2 is also connected to a divider across the 5V rail so that when pin 7 of IC1 goes high, the voltage at pin 2 will be pulled up to around two thirds of the 5V supply, or 3.3V.

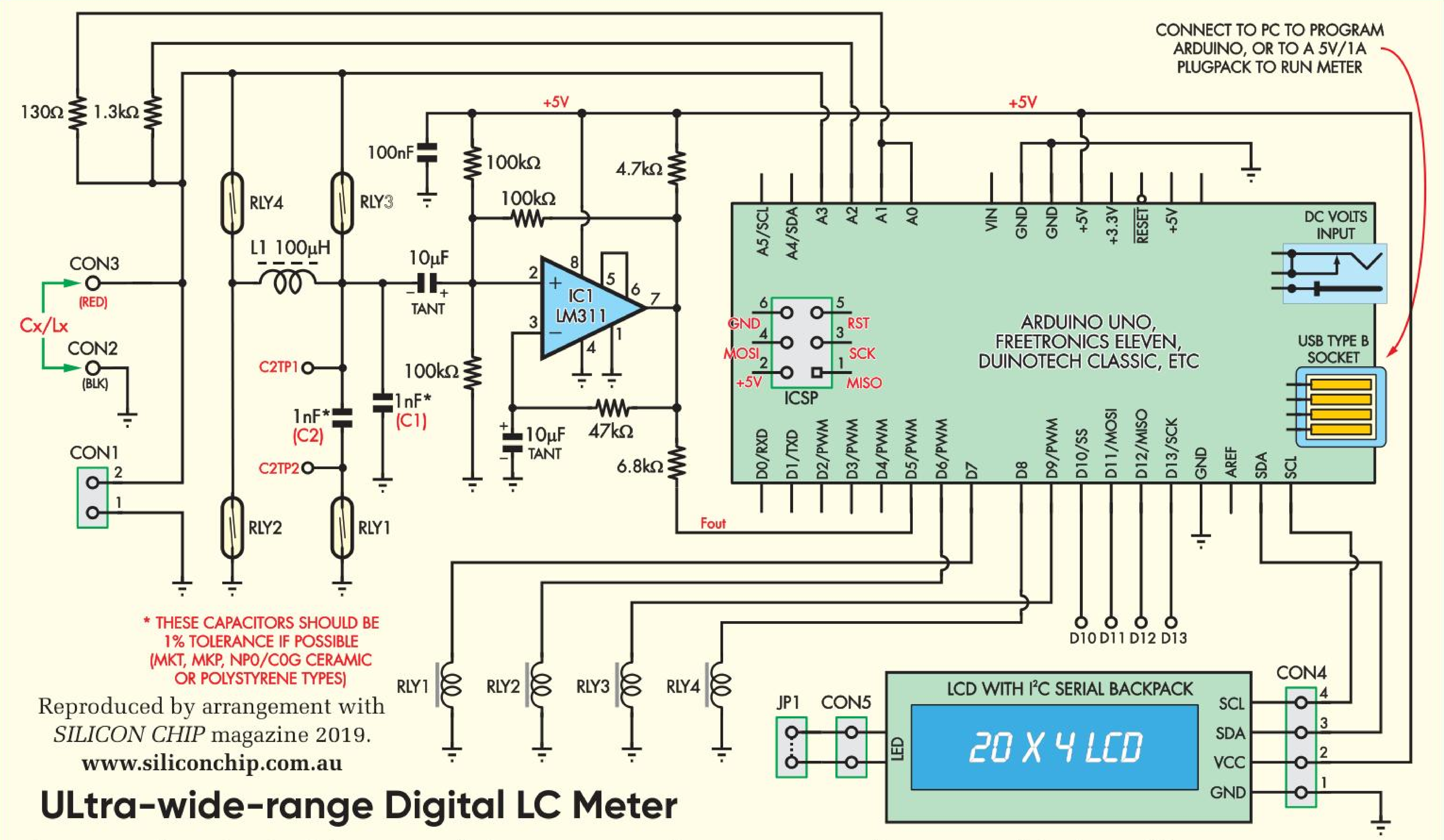
Since IC1's output is a transistor collector, a 4.7kΩ pull-up resistor is used to take it to 5V when that transistor switches off, and it goes low to 0V, when the transistor switches on.

The pin 3 inverting input of IC1 is connected to the output via an *RC* low-pass filter (47kΩ/10µF) and the capacitor charges up to the average output voltage so that the oscillator should stabilise with a reasonably symmetrical waveform.

This waveform is fed to digital input pin D5 of the Arduino via a 6.8kΩ current-limiting resistor.



The three boards which make up the new *LC Meter:* on top is a 4-line alphanumeric display; in the centre is the new Arduino Shield PCB while at the bottom is the Arduino UNO itself (or equivalent).


CONNECT TO PC TO PROGRAM ARDUINO, OR TO A 5V/1A PLUGPACK TO RUN METER
130Ω
1.3kΩ
+5V
+5V
100nF
100kΩ
4.7kΩ
100kΩ
RLY4
RLY3
L1 100µH
10µF
TANT
CON3
(RED)
Cx/Lx
CON2
(BLK)
IC1
LM311
100kΩ
C2TP1
1nF*
(C2)
1nF*
(C1)
10µF
TANT
47kΩ
6.8kΩ
C2TP2
CON1
RLY2
RLY1
Fout
A5/SCL
A4/SDA
A3
A2
A1
A0
VIN
GND
GND
+5V
+3.3V
RESET
+5V
DC VOLTS INPUT
GND
RST
MOSI
SCK
+5V
MISO
ICSP
ARDUINO UNO, FREETRONICS ELEVEN, DUINOTECH CLASSIC, ETC
USB TYPE B SOCKET
D0/RXD
D1/TXD
D2/PWM
D3/PWM
D4/PWM
D5/PWM
D6/PWM
D7
D8
D9/PWM
D10/SS
D11/MOSI
D12/MISO
D13/SCK
GND
AREF
SDA
SCL
D10 D11 D12 D13
* THESE CAPACITORS SHOULD BE 1% TOLERANCE IF POSSIBLE (MKT, MKP, NP0/C0G CERAMIC OR POLYSTYRENE TYPES)
RLY1
RLY2
RLY3
RLY4
JP1
CON5
LED
LCD WITH I²C SERIAL BACKPACK
20 X 4 LCD
SCL
SDA
VCC
GND
CON4

Reproduced by arrangement with *SILICON CHIP* magazine 2019.
www.siliconchip.com.au

ULtra-wide-range Digital LC Meter

Fig.1: complete circuit of the new *Arduino LC Meter*. The component to be measured is plugged into CON1 or CON2/CON3 which is then connected to the test oscillator by RLY3 or RLY4, depending on its type. The Arduino can also perform time-constant-based value measurements using its A0-A3 pins and the 130Ω and 1.3kΩ resistors.

Oscillator configurations

Four reed relays (RLY1-4); each connect to either one end of L1 or to the calibration capacitor C2 and are used by the micro to select one of five different modes. Each relay's state in each mode is shown in Table.1. The modes work as follows.

Mode 1

This Mode is for oscillator calibration. RLY2 is energised and its contacts are closed, effectively connecting L1 and C1 in parallel. The other relays are not energised. L1 and C1 resonate and cause IC1's output to produce a square wave of around 500kHz, which can be measured by the Arduino using its internal timer hardware.

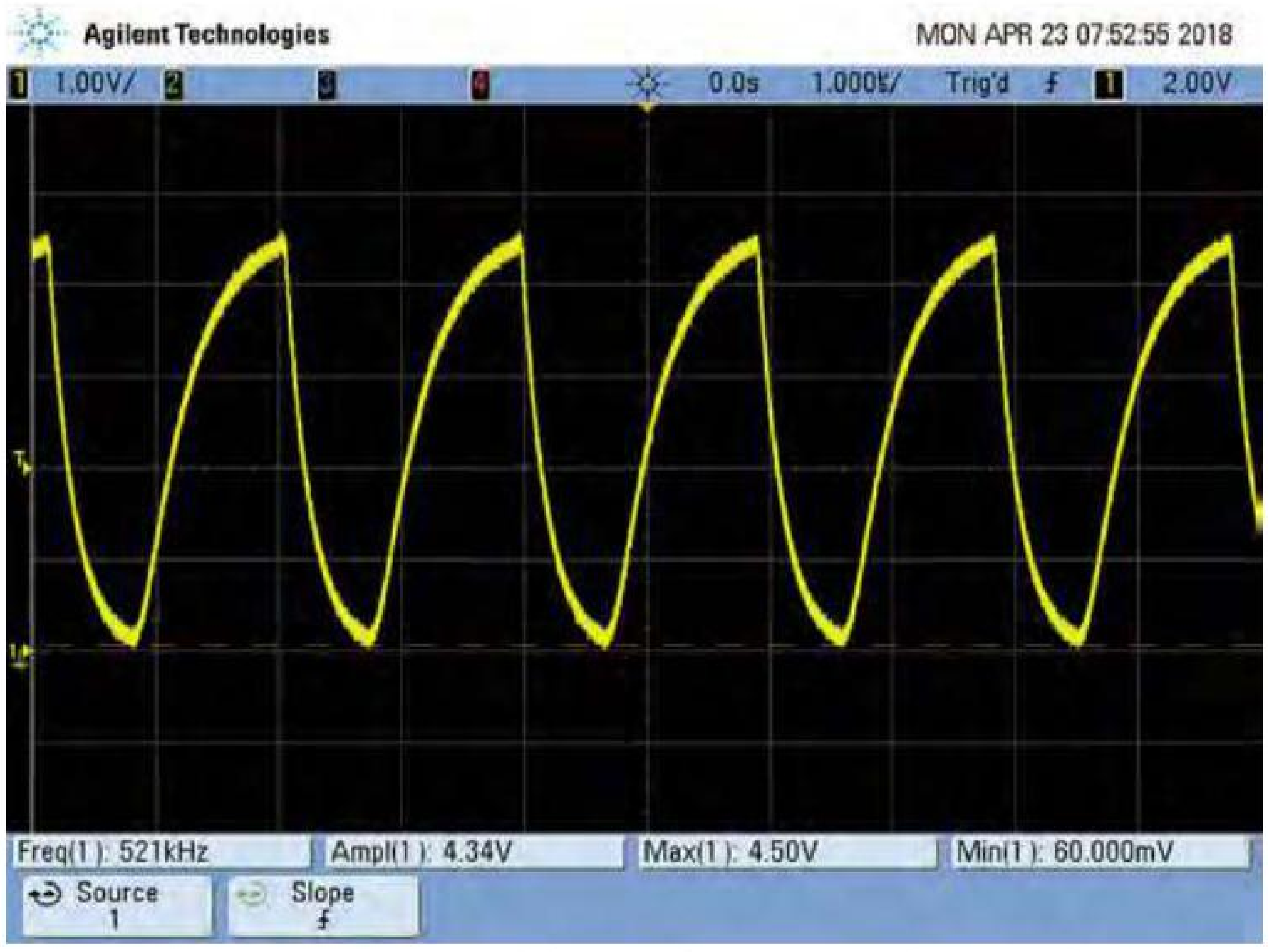
Mode 2

The second Mode is also for calibration. Both RLY1 and RLY2 are now energised but RLY3 and RLY4 are not. This is identical to Mode 1 except that now, C2 is connected in parallel to C1. The oscillator frequency drops to around 370kHz, due to the doubled capacitance. This allows the unit to measure the stray capacitance on the PCB and provide more accurate component value measurements.

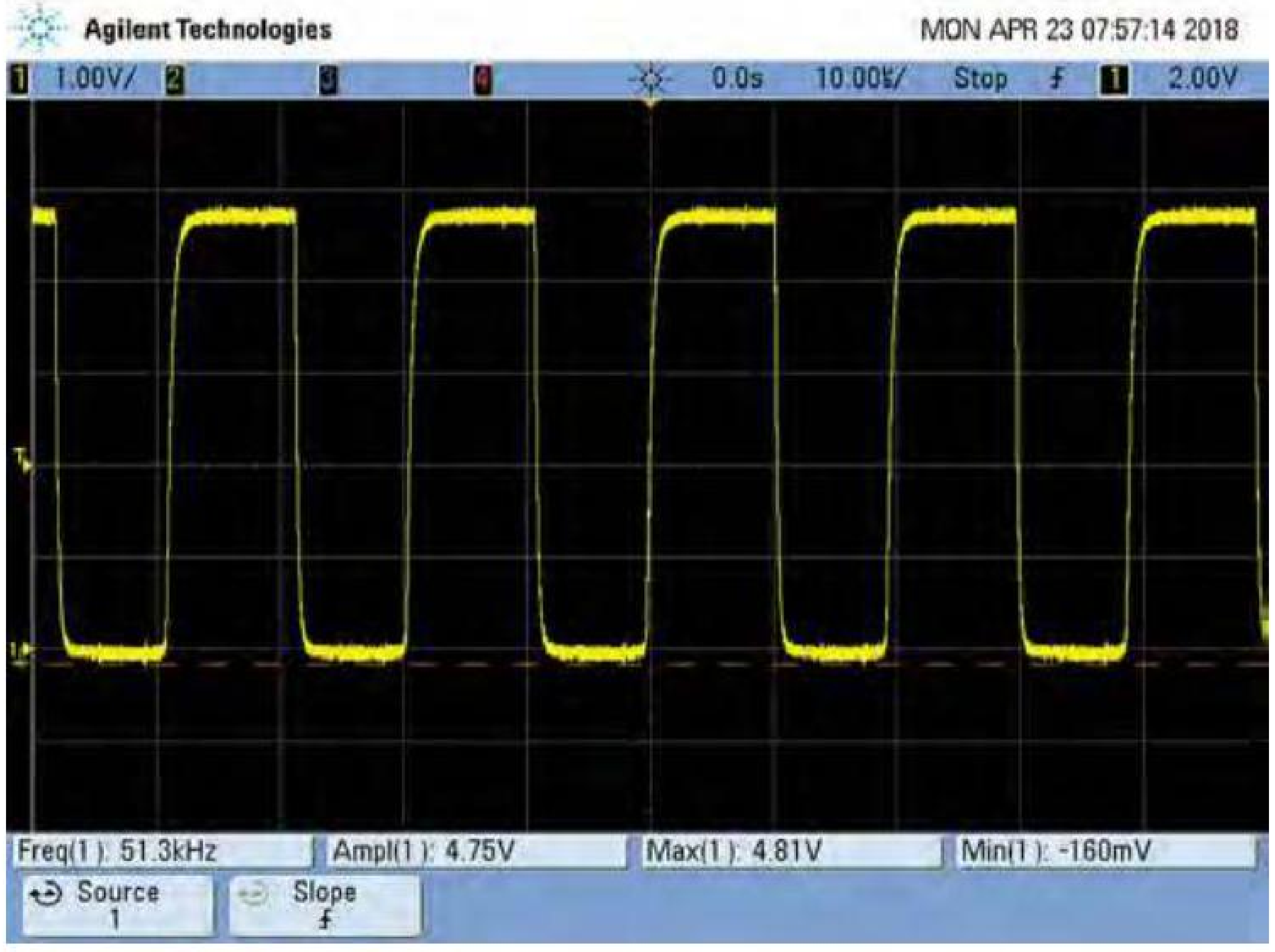
Mode 3

Next, there is Mode 3 for measuring the value of a capacitor connected either between the pins of CON1 or between banana sockets CON2 and CON3 (these are effectively in parallel). In this case, RLY2 and RLY3 are energised and RLY1 and RLY4 are not. This is similar to the first mode except that now the DUT is connected in parallel with C1.

Since the capacitance has increased, this should result in a lower oscillator frequency and by measuring the change, we can calculate the capacitor value, using the method explained previously.


Agilent Technologies
MON APR 23 07:52:55 2018
1 1.00V/
0.0s
1.000µ/
Trig'd
1 2.00V
Freq(1): 521kHz
Ampl(1): 4.34V
Max(1): 4.50V
Min(1): 60.000mV
Source 1
Slope

Scope1: measurement of the F1 frequency during the calibration phase. As detected by the scope, the frequency is 521kHz with an amplitude of 4.34V.


Agilent Technologies
MON APR 23 07:57:14 2018
1 1.00V/
0.0s
10.00µ/
Stop
1 2.00V
Freq(1): 51.3kHz
Ampl(1): 4.75V
Max(1): 4.81V
Min(1): -160mV
Source 1
Slope

Scope2: measurement of a 100nF capacitor. The frequency has dropped to 51.5kHz. Note the amplitude is up to 4.75V, and the oscillator seems quite stable.

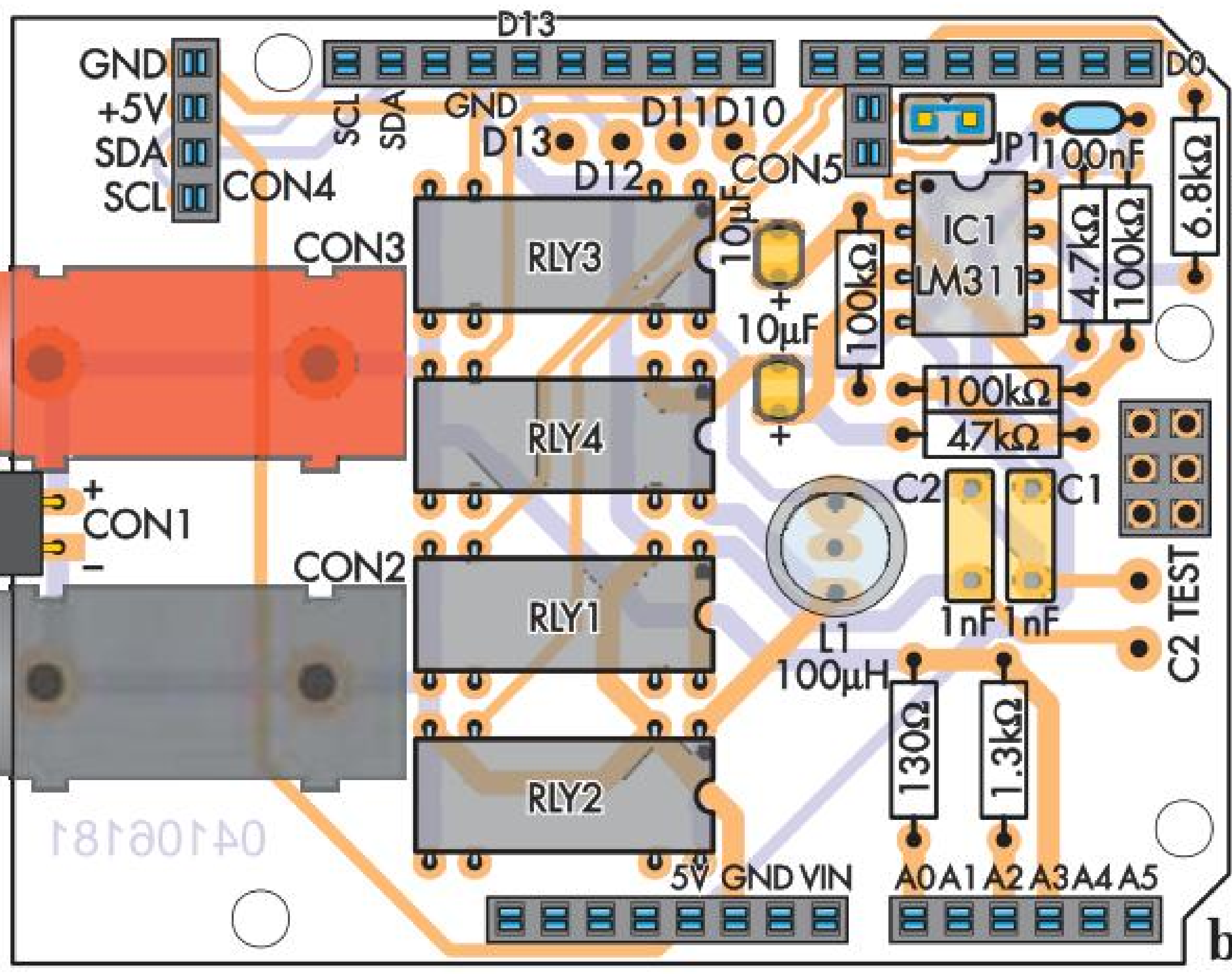
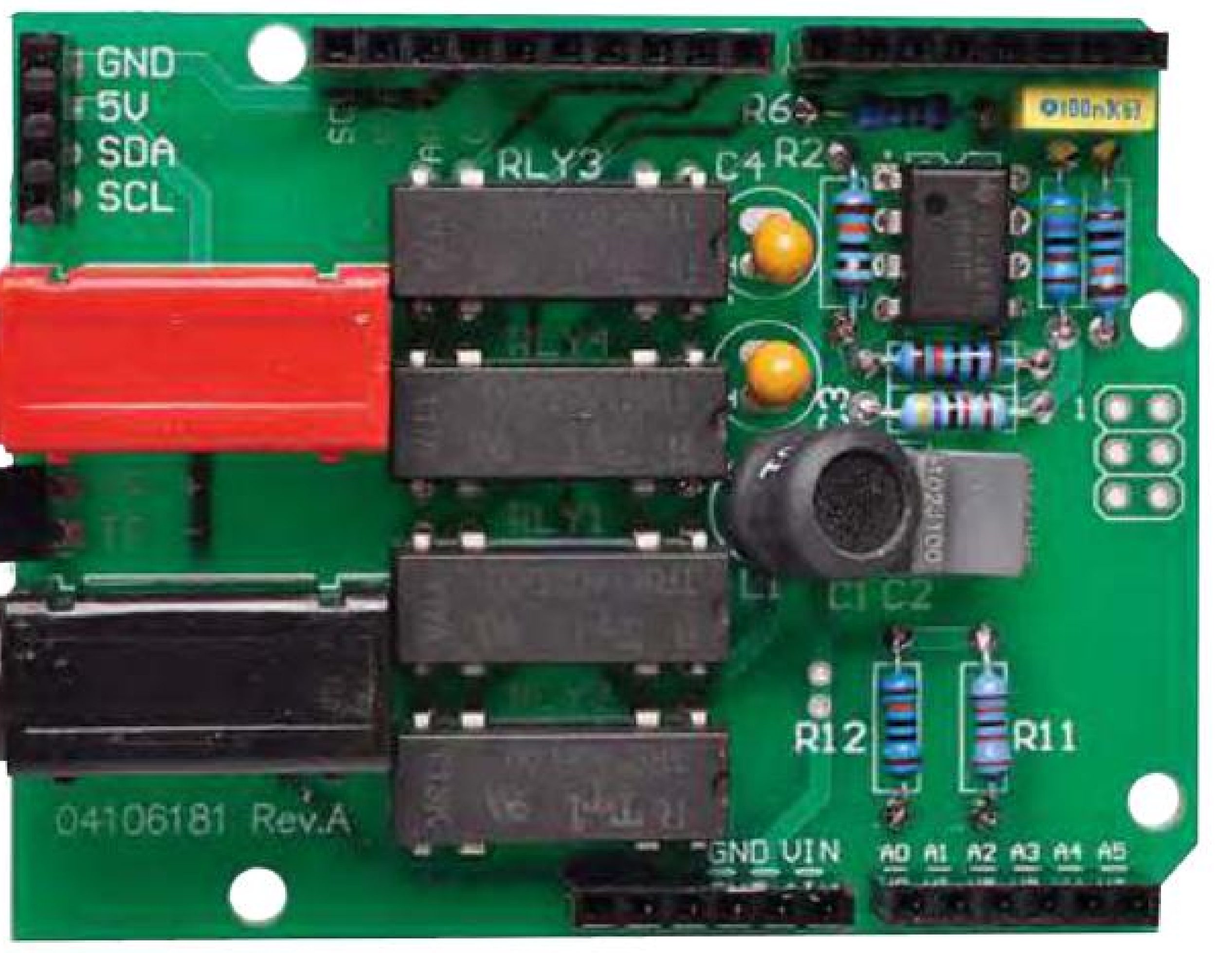

GND
+5V
SDA
SCL
CON4
CON3
RLY3
RLY4
CON1
CON2
RLY1
RLY2
D13
SCL
SDA
GND
D13
D12
D11 D10
CON5
JP1
100nF
D0
6.8kΩ
IC1
LM311
4.7kΩ
100kΩ
10µF
100kΩ
47kΩ
C2
C1
1nF 1nF
L1
100µH
130Ω
1.3kΩ
C2 TEST
5V GND VIN
A0 A1 A2 A3 A4 A5
04106181

Fig.2: follow this overlay diagram – and the photo at right – to build the *LC Meter* shield PCB. Be careful with the placement of L1 as an extra pad is provided for larger inductors; also ensure that RLY1-RLY4, IC1 and the two tantalum capacitors are fitted with the orientation shown. The PCB at right is an early prototype; some minor adjustments have been made in the final version.


GND
5V
SDA
SCL
RLY3
C4
R2
R6
L1
C1 C2
R12
R11
04106181 Rev.A
GND VIN
A0 A1 A2 A3 A4 A5

Mode 4

This is for measuring inductance. In this case, only RLY4 is energised. The DUT is connected in series with L1 and its opposite end is connected to ground via CON1 or CON2.

This means that the resonant circuit inductance has effectively increased (by the value of the DUT) and so once again, the oscillator frequency should drop and the difference can be used to calculate the inductance of the unknown component.

As explained earlier, in the above modes, the A0-A3 pins on the Arduino are kept in a high impedance state so they won't interfere with the oscillator. Their only influence is in their (small) pin capacitance and this is compensated for during calibration.

Mode 5

Finally, Mode 5 is used for measuring high values of inductance and capacitance, and now the A0-A3 pins become active.

In this mode, all four relays are off and the DUT is not connected to the oscillator. Instead, the 130Ω and 1.3kΩ resistors are used to drive the DUT and A0, A1 and A2 become outputs at different times.

A0 and A1 are connected together to measure the internal pin resistance, because this appears in series with the 130Ω resistor and can cause measurement errors.

The A3 pin is used as an analogue input, to measure how the voltage across the DUT changes in response to the current from the other two pins.

The two different resistors value are used to provide two different ranges, to improve the unit's accuracy. In this mode, capacitance and inductance measurements are based on charge time measurement, as described below.

You don't have to select any of these modes or tell the micro that you are measuring capacitance or inductance. The micro does it automatically.

Displaying the results

Results are displayed on a 20×4 alphanumeric LCD that is fitted with an I^{2}C adaptor so that only four connections to the Arduino are required: +5V power and ground, and the SDA and SCL pins for I^{2}C communications.

The four-line display constantly shows all the measurement data that you need to see and it even keeps a running average for super-accurate measurements.

It's powered from the Arduino's 5V rail, which can be derived from a USB charger, PC USB port or 9-12V DC plugpack via the on-board barrel connector.

The four reed relay coils are driven directly from the Arduino's digital output pins D6-D9. These outputs can provide more than enough current to latch a reed relay (up to 40mA) and the back-EMF at switch-off is sufficiently low that the ATmega328 IC's internal clamp diodes are sufficient for absorbing it.

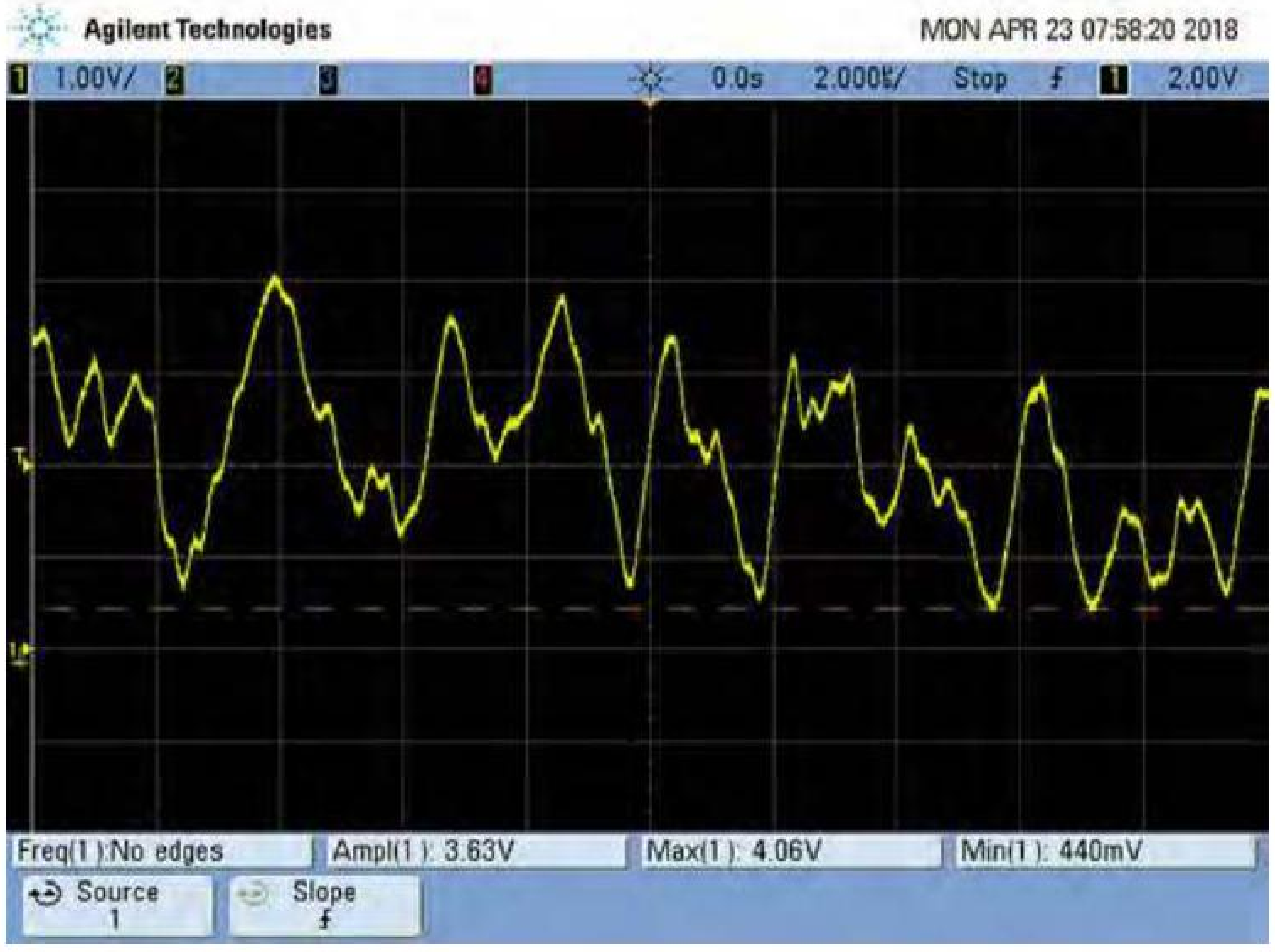
By the way, don't be tempted to substitute a different type of Arduino board. You need to use the Uno or equivalent. This is because the *Frequency Counter* library that we use depends on pin D5 being fed into one of the hardware timers and this is only the case with the Uno.

Construction

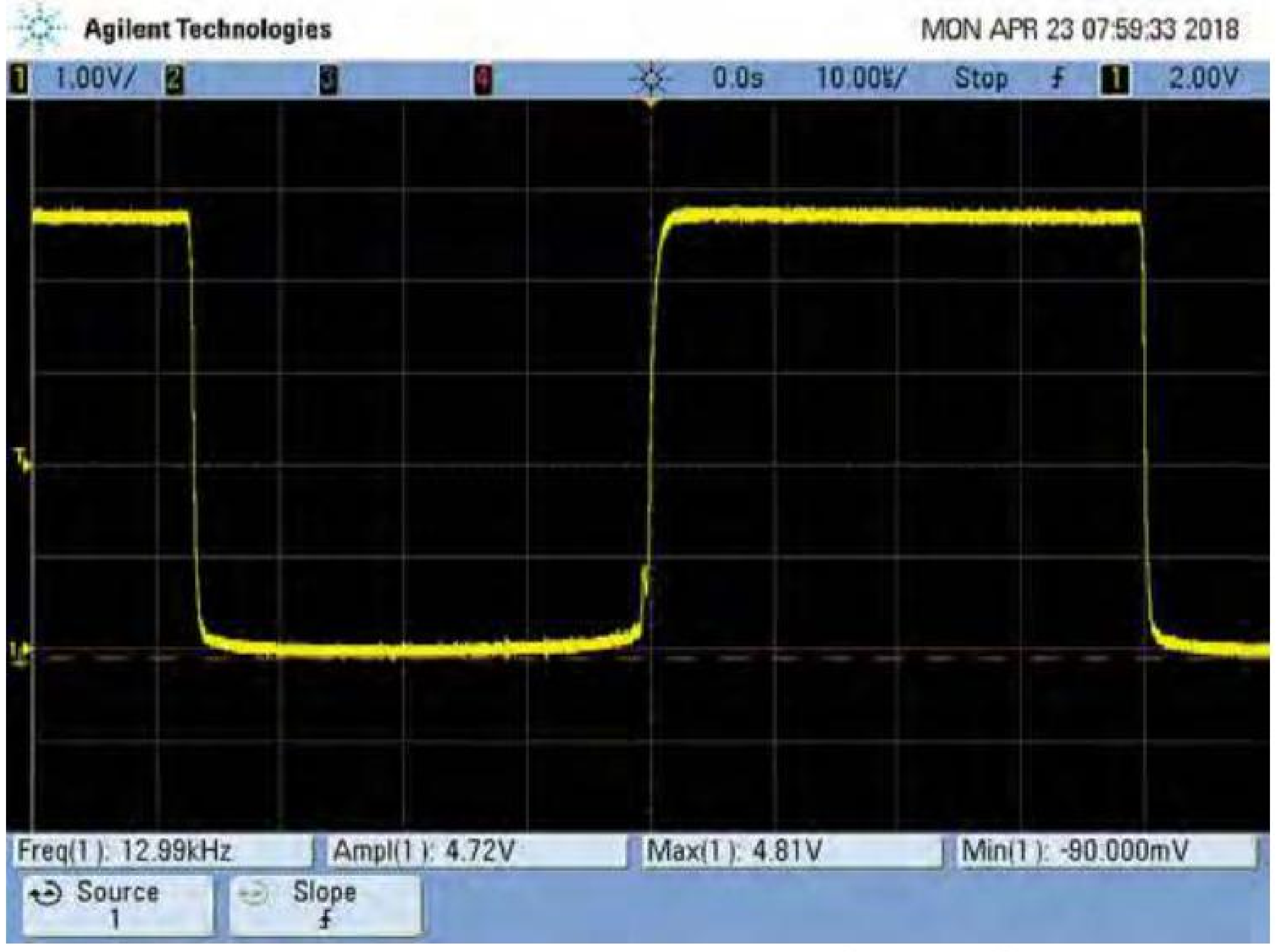
The custom shield is built on a double-sided PCB measuring 68.5 x 53mm (standard shield size), and coded

Mode	RLY1	RLY2	RLY3	RLY4
1. Calibration (C1 only)	OFF	ON	OFF	OFF
2. Calibration (C1 and C2)	ON	ON	OFF	OFF
3. Capacitor measurement	OFF	ON	ON	OFF
4. Inductor measurement	OFF	OFF	OFF	ON
5. Time constant mode	OFF	OFF	OFF	OFF

Table.1: relay states in each test mode


Agilent Technologies
MON APR 23 07:58:20 2018
1.00V/
0.0s
2.000s/
Stop
2.00V
Freq(1):No edges
Ampl(1): 3.63V
Max(1): 4.06V
Min(1): 440mV
Source 1
Slope

Scope3: in this case the oscillator is not stable when measuring a 407µF capacitor. To get an accurate measurement you need to use the *RC* method (see Scope6).


Agilent Technologies
MON APR 23 07:59:33 2018
1.00V/
0.0s
10.00s/
Stop
2.00V
Freq(1): 12.99kHz
Ampl(1): 4.72V
Max(1): 4.81V
Min(1): -80.000mV
Source 1
Slope

Scope4: measuring a 1.5µF capacitor. The oscillator appears stable, but a small glitch appears at the start of the leading edge.

Parts list – wide range, Arduino-based LC Meter

1 double-sided PCB, 68.5 x 53mm [*PE PCB Service* code 04106181]
1 set of Arduino stackable headers (1x6 pin, 2x8 pin, 1x10 pin, Jaycar HM3208)
1 Arduino Uno or equivalent [Altronics Z6280, Jaycar XC4410]
1 20×4 alphanumeric LCD [Jaycar QP5522]
1 I^2C Port Expander module [Jaycar XC3706]
1 100μH bobbin-style inductor [Altronics L6222, Jaycar LF1102]
4 5V coil DIL reed relays [Altronics S4100, Jaycar SY4030]
2 2-pin female header sockets (CON1,CON5)
1 black PCB-mounting right-angle banana socket (CON2) [Altronics P9201]
1 red PCB-mounting right-angle banana socket (CON3) [Altronics P9200]
1 4-pin female header socket (CON4)
1 2-pin header with shorting block (JP1)

Semiconductors
1 LM311 high-speed comparator, DIP-8 [Altronics Z2516, Jaycar ZL3311]

Capacitors
2 10μF 6.3V tantalum
1 100nF MKT or ceramic *(code 0.1µf; 104 or 100n)*
2 1nF ±1% polypropylene (MKP) or C0G/NP0 ceramic capacitor or polystyrene [SC shop SC4273] *(code 0.001µf; 102 or 1n)* [Mouser 80-PFR5102F100J11L4B]

Resistors (all 0.25W, 1% metal film)
3 100kΩ
1 47kΩ
1 6.8kΩ
1 4.7kΩ
1 1.3kΩ
1 130Ω

Case components (if required – see text)
1 laser-cut clear acrylic case (6 pieces) [www.siliconchip.com.au/Shop/19/4609]
4 small self-adhesive rubber feet
2 M3 x 15mm nylon machine screws
4 M3 nuts
2 M3 x 32mm machine screws
2 M3 nylon washers
4 M3 nylon nuts
4 M3 x 25mm machine screws
2 M3 x 15mm tapped spacers

04106181. It is avaiable from the *PE PCB Service*.

The overlay diagram is shown in Fig.2 – use this and the matching photo as a guide during construction.

Start by fitting the resistors. While their values are printed on colour-coded bands, it's safer to simply measure the values with a DMM before soldering them in place. Next, mount the MKT/ceramic capacitors and inductor L1. None of these components are polarised.

What you wouldn't give to have an accurate *LC Meter* on hand right now! While there are three holes for L1, only two are needed; the extra hole is to allow for variations in component size.

Ensure that one of the inductor leads goes through the hole closest to to the bottom edge of the PCB. The other lead can be soldered to either of the other pads.

Now fit the two tantalum capacitors. These are polarised; their positive leads will be identified with a '+' symbol printed on their bodies. Ensure the lead on this side goes to the pad marked '+' on the PCB; ie, closer to inductor L1.

We recommend that you solder IC1 directly to the board (ie, don't use an IC socket) to avoid socket contact resistance. Ensure the notched end is facing the top of the board before soldering it in place, as shown in Fig.2.

Now mount reed relays RLY1-RLY4. They are all identical and all have the same orientation. While they are in DIL packages, they should be soldered directly to the PCB. Again, this is to avoid socket contact resistance.

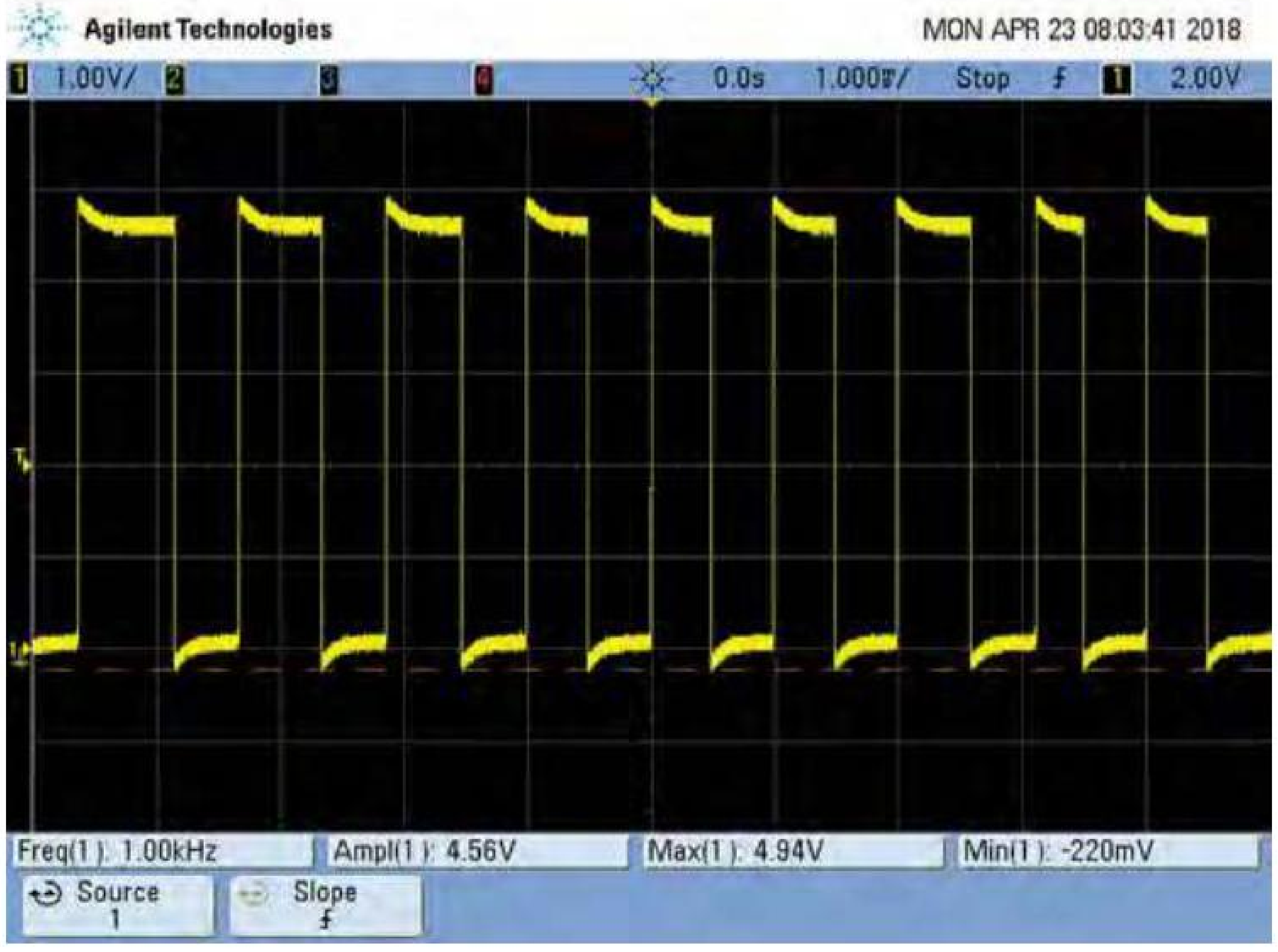
In each case, pin 1 faces towards the right-hand end of the PCB, away from the test sockets.

SIL socket CON1 is provided to allow small components to easily be tested as their leads can simply be pushed into the socket spring contacts. Take a two-way header socket and bend its pins by 90° close to the socket body, then solder it so that the socket projects off the end of the PCB. You can now solder the two banana sockets in place; CON3 is red while CON2 is black.

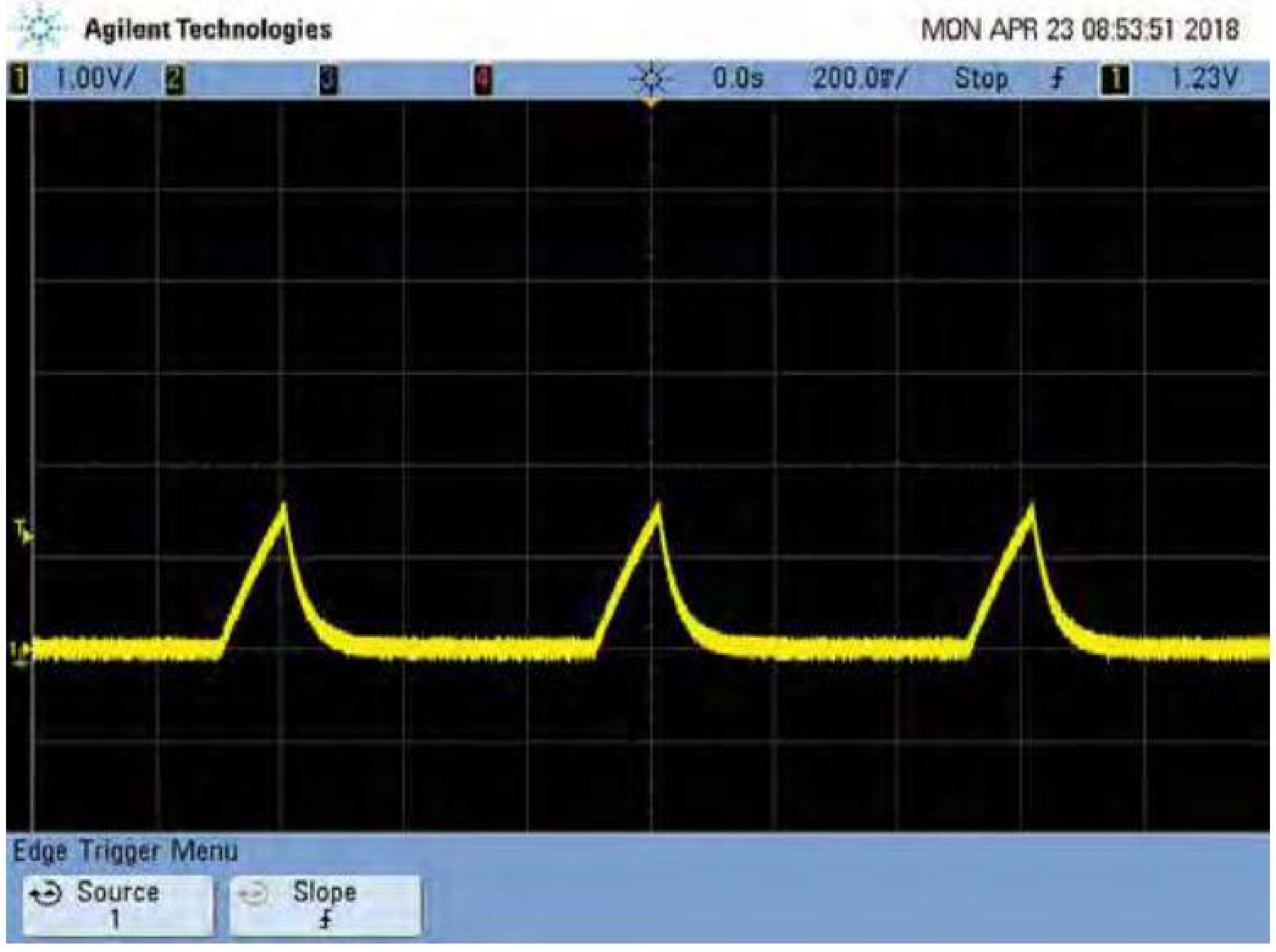
Mount the four-way and two-way header sockets for CON4 and CON5 next, along with the pin header for JP1. You can use the I^2C module board as a jig to line up the sockets correctly. You may also need to straighten out the pins on the I^2C breakout board, so they are facing downwards.

The final step in the shield construction is installing the headers to connect the shield to the Uno. We have used stackable headers, as they are slightly longer, giving some clearance between the Uno's USB socket and the TP+ test point.

The easier way to install the headers accurately is to assemble the headers,



Scope5: measuring a 0.5H (approximate) inductor. The frequency is down to 1kHz, but the oscillator is less stable – note that the pulse widths are varying significantly.



Scope6: a 220µF capacitor in *RC* mode. The measurement is repeated (about every second depending on the capacitor value) and the capacitor is discharged at the end of each cycle, to be ready for the next cycle.

shield and Uno together and use the Uno's headers to line up the shield headers. Then turn it all upside down so that the shield rests on the headers as far away from the Uno as possible.

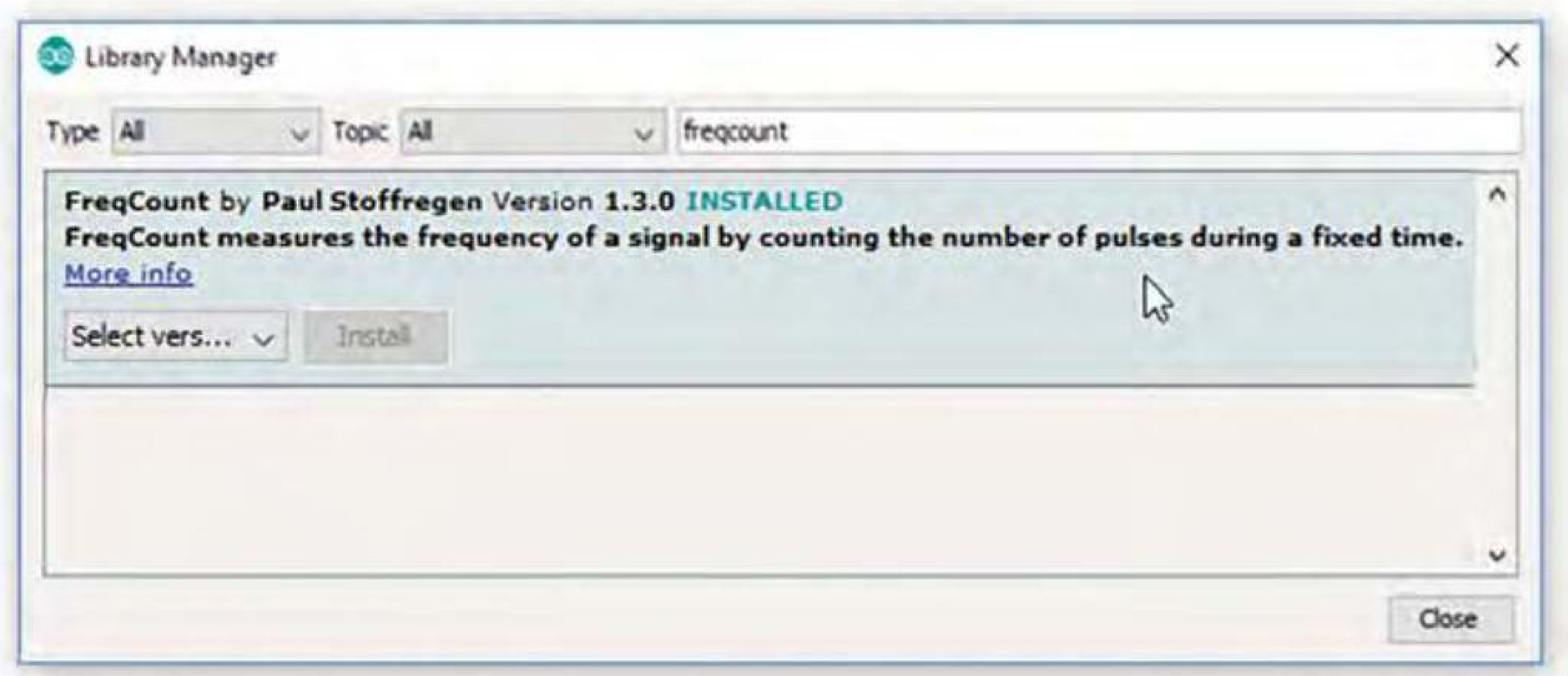
Tack solder the corner pins in place, then remove the Uno main board to allow easier access to the rest of the pins for soldering. Then refresh the corner pin solder joints.

If you don't already have the I^{2}C breakout board fitted to the LCD, carefully line up pin 1 of the LCD with the I^{2}C header end of the adaptor.

Solder one pin, then confirm that the boards are parallel and straight but not touching anywhere except the header before soldering the rest of the pins. The LCD assembly can now be plugged into CON4 and CON5. Alternatively, run four breadboard jumper leads between the I^{2}C adaptor and CON4 for testing.

If you haven't yet adjusted the contrast on the LCD, this is easier to do when the board is connected by jumper leads.

Finally, push the assembled shield onto the Uno. This completes construction. Plug the Uno into your computer using a standard USB cable.



FreqCount by **Paul Stoffregen** Version **1.3.0** **INSTALLED**
FreqCount measures the frequency of a signal by counting the number of pulses during a fixed time.
More info

Fig.3: this shows how to find and install the *Frequency Counter* library in the Arduino IDE Library Manager. Type `freqcount` in the box at the upper right-hand corner of the window and then click the Install button that appears below (it's grey here because we have already installed it).

Loading the software

To compile and upload the software that runs on the Uno board, you need to have the Arduino IDE (Integrated Development Environment) software installed on your computer. The IDE can be downloaded from **www.arduino.cc/en/main/software** and it is available for Windows, macOS X and Linux. Download and install a version to suit your operating system and start the software.

The 'sketch' or program that runs on the Uno needs two external libraries. One is used to count the pulses that are generated by the oscillator and the other to interface to the I^{2}C LCD.

They are both supplied as ZIP files in the download package, along with the sketch itself.

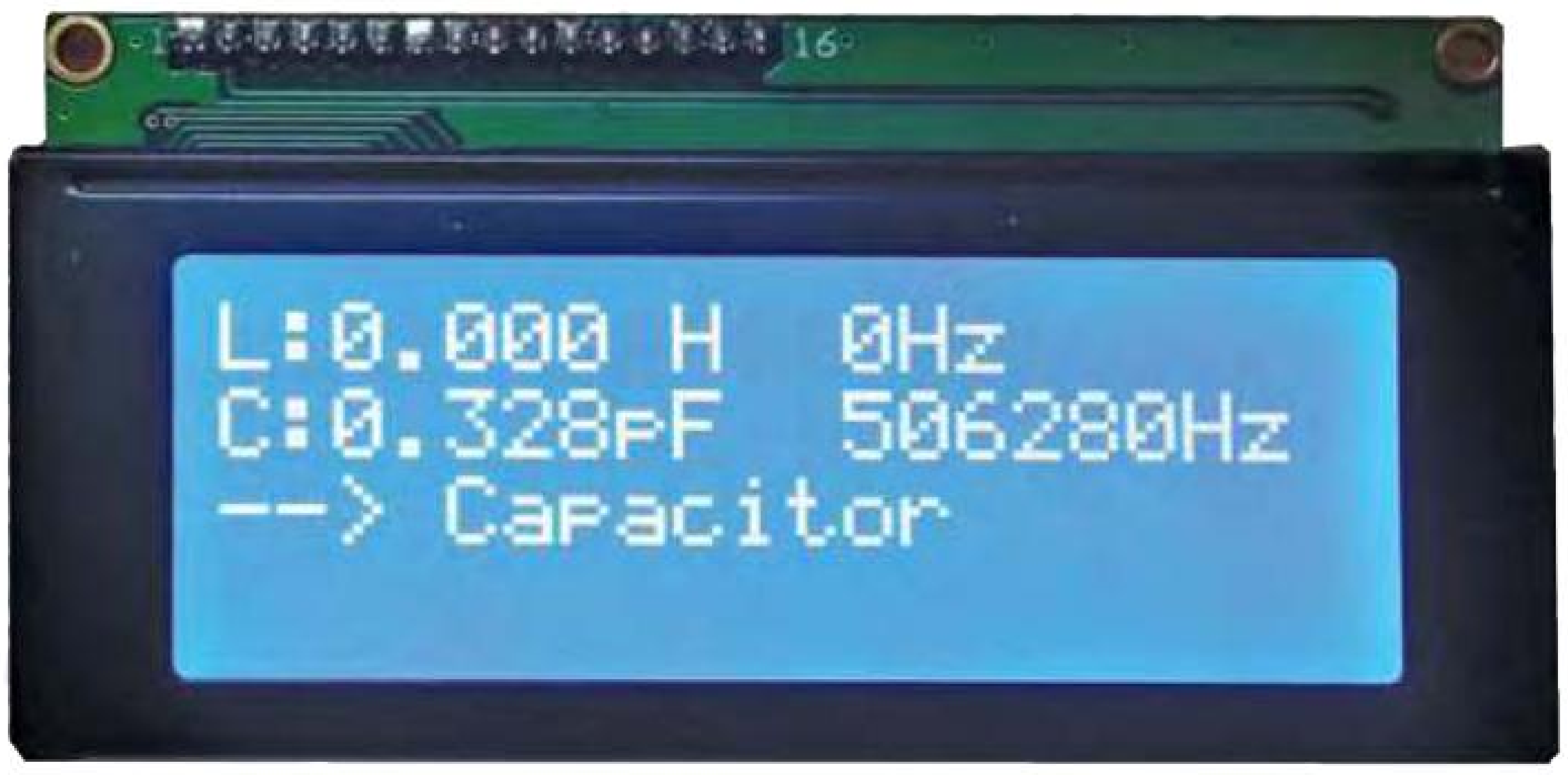
Installing the first library is as simple as going to the Sketch -> Include Library -> Manage Libraries... menu, and searching for `freqcount`, and clicking on the install option that is presented (see Fig.3).

To install the second library, search for `liquidcrystal_pcf8574` in the Library Manager, and install the version by Matthias Hertel.

Now open the sketch file, select 'Arduino/Genuino Uno' under the Tools -> Board type... menu and then use the Tools ->Port menu to select the serial port that the Arduino is plugged into. Most versions of the Uno will display as COMx: (Arduino/Genuino Uno) in the dropdown menu, so you can use this hint to find the correct serial port if you are unsure.

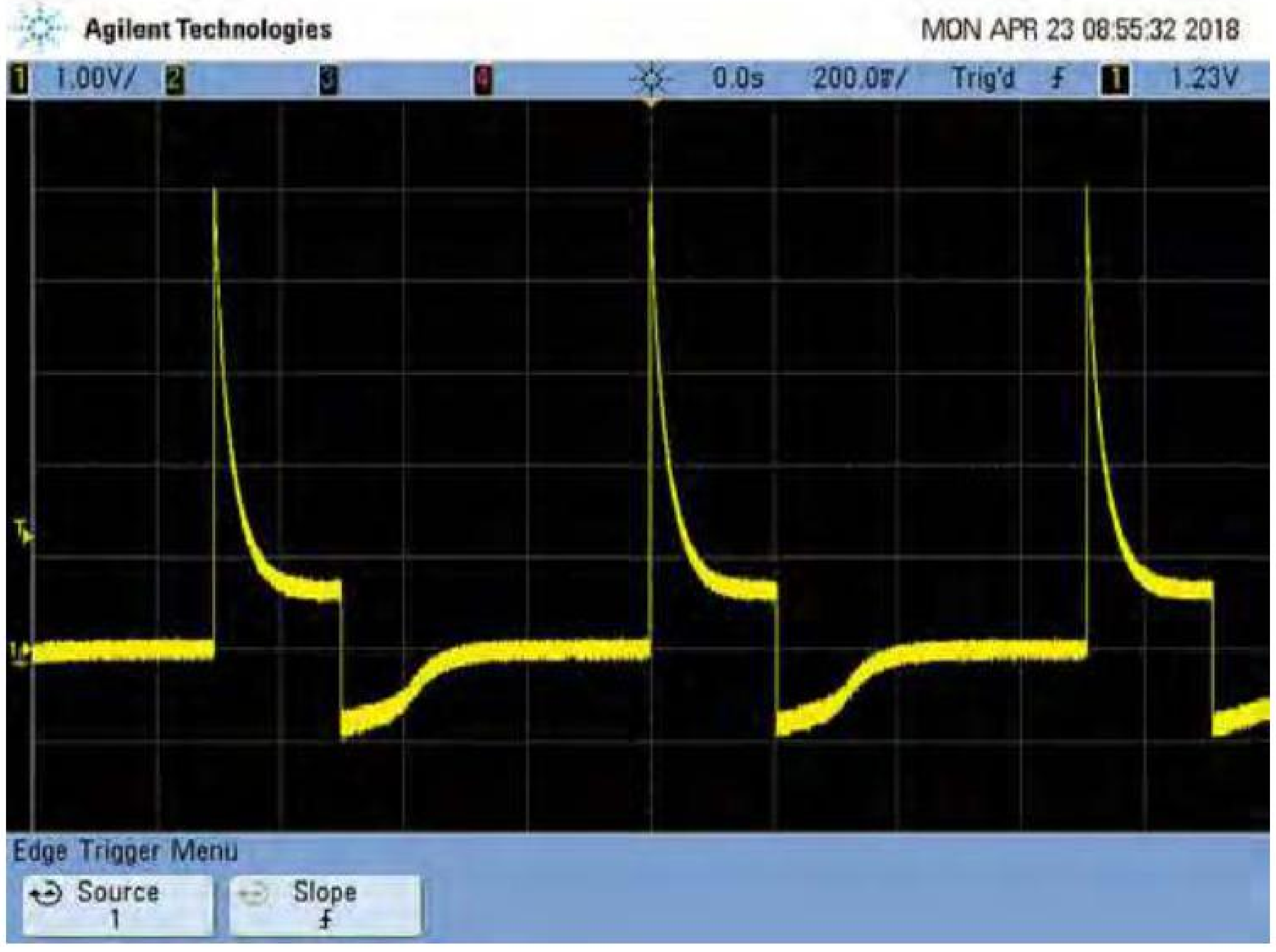
Press Ctrl-U to compile and upload the sketch. If you see the message 'Done Uploading' at the bottom of the window then everything has compiled and uploaded successfully.

If you get an error message, check that the libraries are installed correctly, and check that the correct serial port is selected.

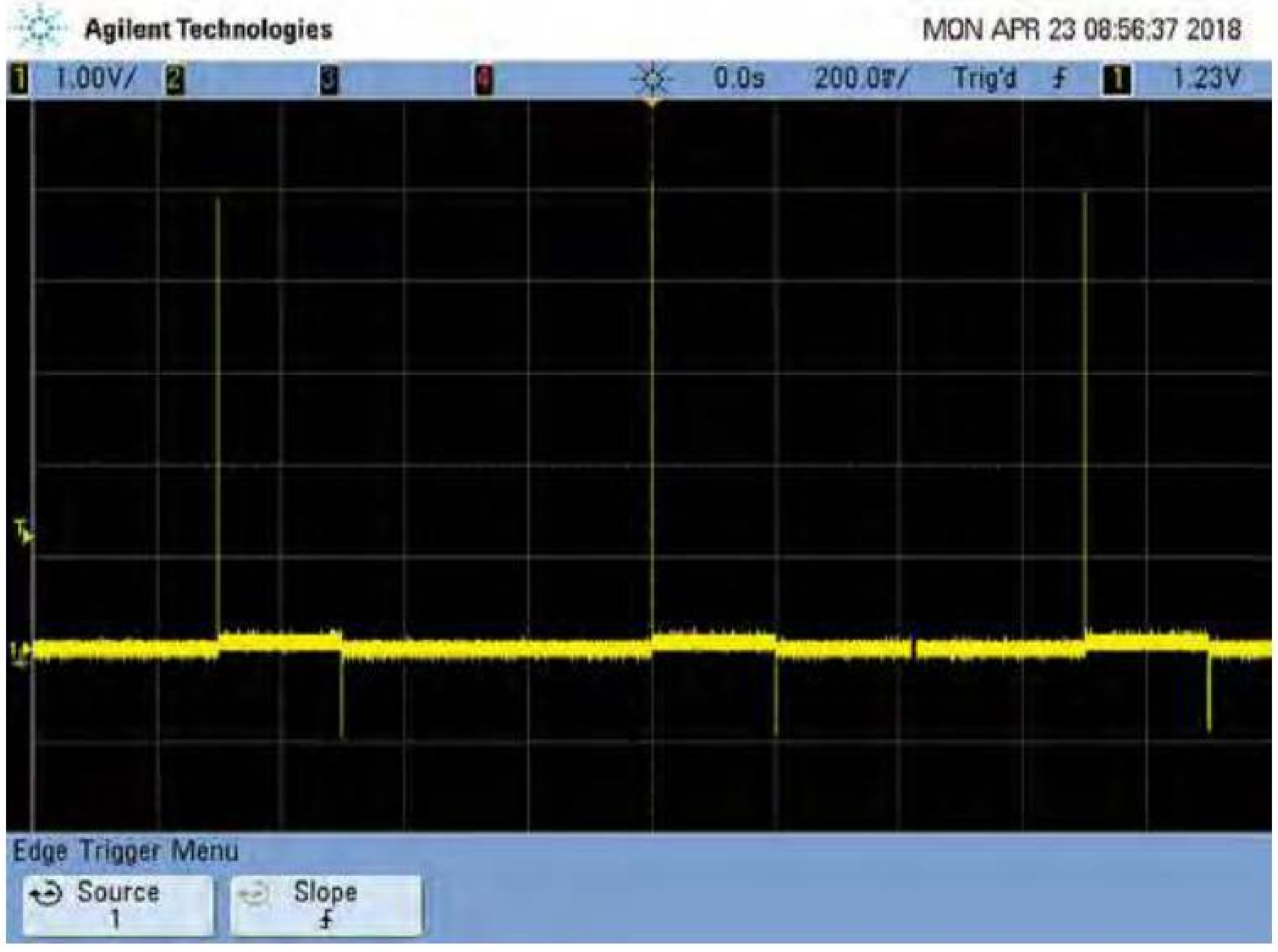


```
L:0.000 H  0Hz
C:0.328pF  506280Hz
--> Capacitor
```

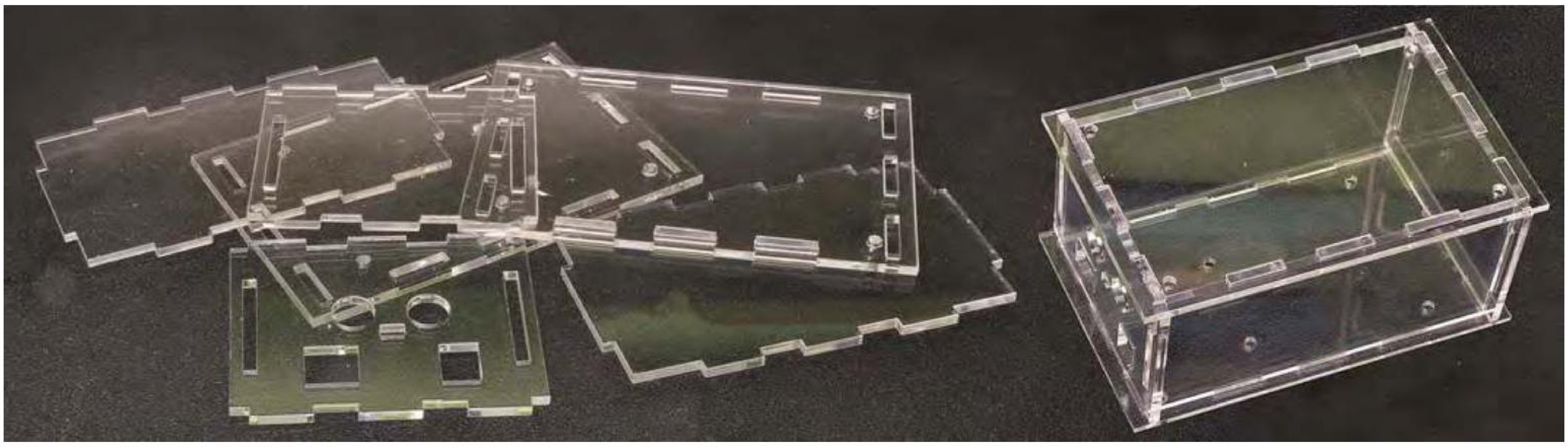
Fig.4: typical display on the *LC Meter* with no component connected. The small amount of residual capacitance shown can be adjusted for during calibration.



Scope7: measuring a 0.5H inductor in *RL* mode. Again, the cycle is repeated about once a second. The effects of the inductor's intrinsic resistance can be seen in that the voltage across the inductor does not fall completely to 0V.



Scope8: measuring a 26mH inductor in *RL* mode. Note that the spikes are very brief (microseconds) and the slightly raised trace indicating the intrinsic resistance of the inductor is significant in this case too.



On the left are the six pieces which make up the case specifically designed for this *LC Meter*. They simply slot together and the PCB mounting screws hold them in place. The assmbled case is shown on the right. No cutout is necessary for the display as the case is clear acrylic. This case is available from the SILICON CHIP Online Shop (Cat SC 4609).

Testing and set-up

If there is no back-light on the LCD, check that the LCD back-light jumper is installed on the I^{2}C breakout board. If you are using the back-light header for mounting, the jumper is installed on the two-pin header next to the mounting header.

If the back-light is working but there is no text, check and adjust the contrast pot on the back of the I^{2}C breakout board. If you have no text or faint text, try turning the pot clockwise. If you can only see white squares, try turning the pot anti-clockwise.

You should see text similar to that shown in Fig.4.

The unit stores calibration data in EEPROM. The first time you power it after uploading the sketch, it will load a sensible set of defaults so you can start using it straight away. In the unlikely case that this does not happen, you can reset the calibration data using the following procedure:

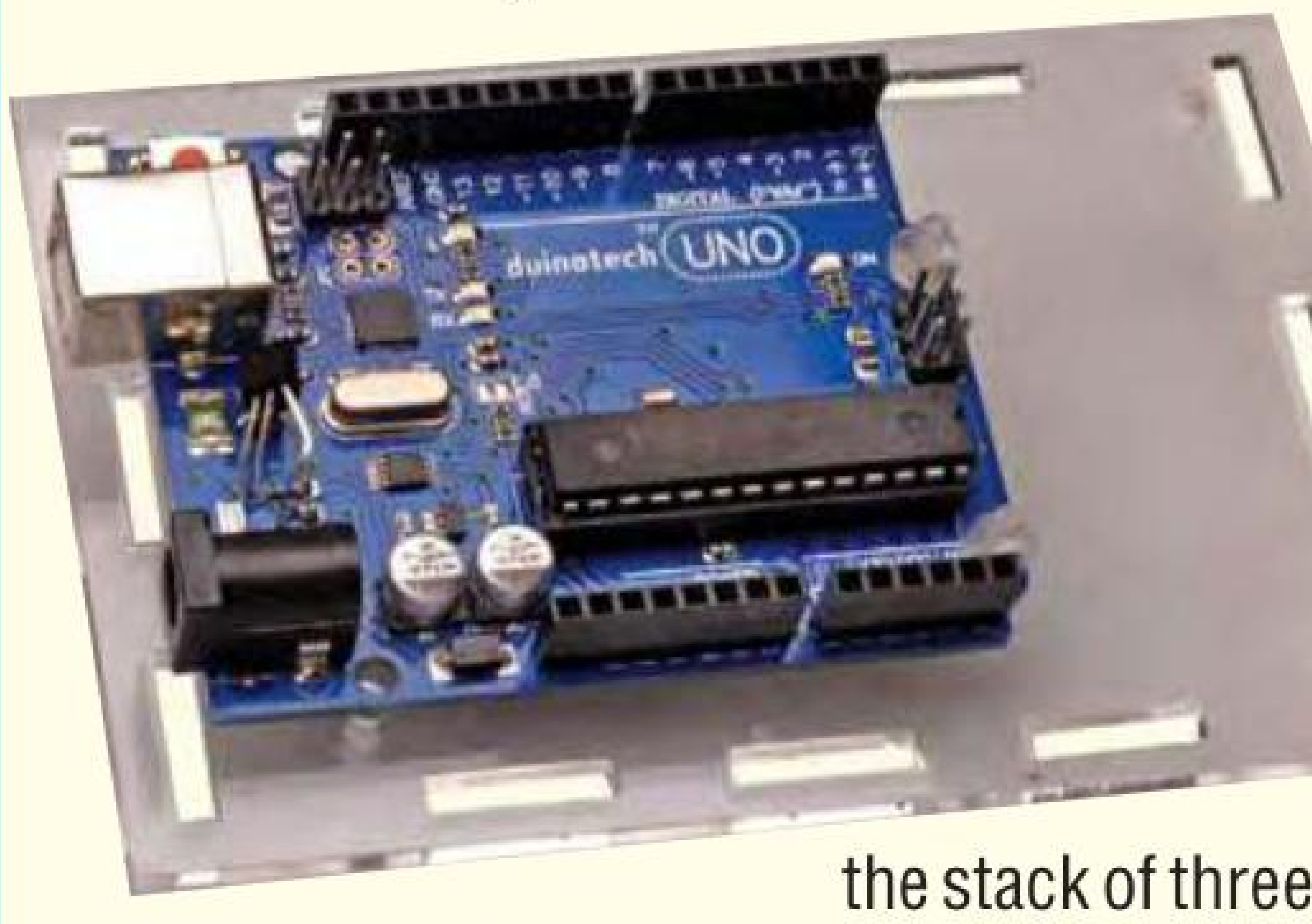
Open the serial monitor at 115,200 baud, and type 'C' followed by Enter. When the menu appears, press 'L', Enter, 'S', Enter, then 'X', Enter, and press the reset button on the Uno.

Now attach a component to the test terminals and check that you get a reading of its value. *Do note that polarised components, such as electrolytic and tantalum capacitors should match their polarity to the test terminal markings.* The test terminals (CON1, CON2/CON3) may have up to 5V present, so take care not to attach any components with lower ratings.

You can improve the accuracy of the meter by measuring and entering specific values into the calibration values (assuming you have the means to do so). The values of C2 and the 130Ω and 1.3kΩ resistors are initially assumed to be very close to expected. If they're not exact, and you can measure them accurately, you can improve the unit's accuracy by entering these into the calibration data, as we'll explain later.

Assembly in the acrylic case

If you want to use the optional purpose-designed clear acrylic case, it is available from the SILICON CHIP Online Shop at **siliconchip.com.au/shop** – cat no is SC-4609; the price is very reasonable too!).

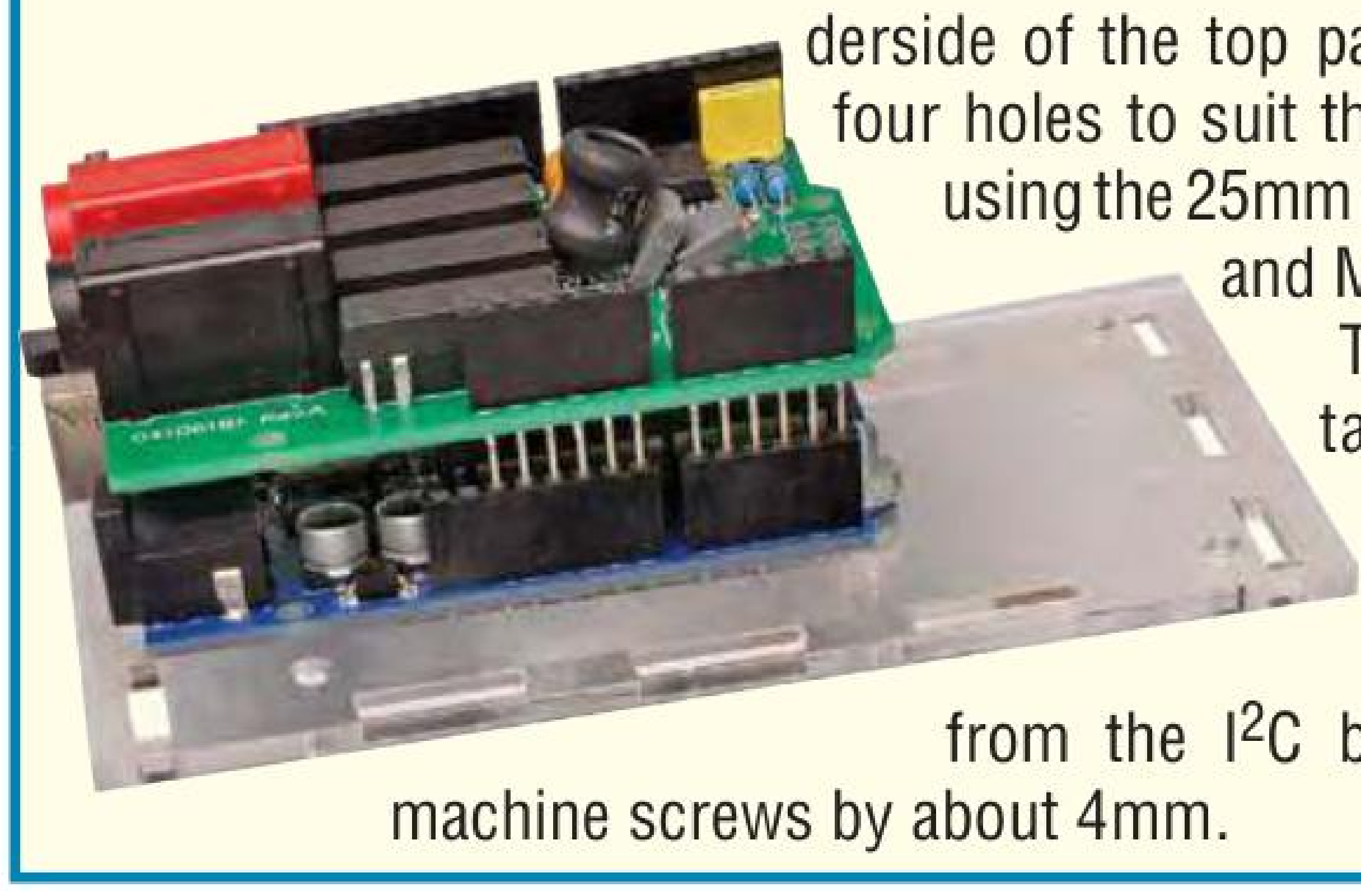


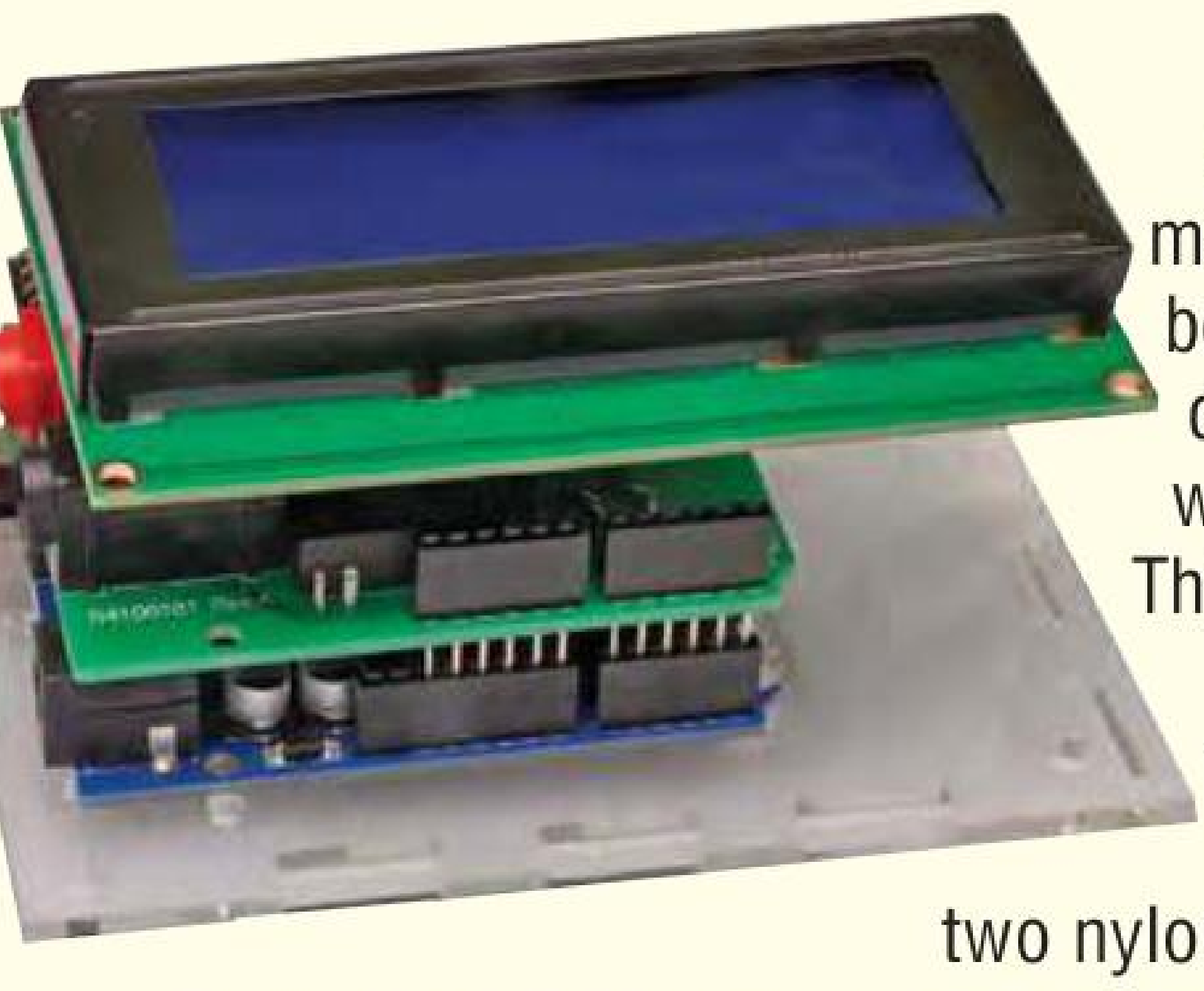
This case makes assembly so much simpler. It simply slots together. And it looks really professional!

You *could* use a jiffy box like our earlier LC Meters but the stack of three PCBs does not easily lend itself to a UB3 case – and you also have a number of slots/holes to prepare. This case has all required slots and holes already prepared for you.

Carefully remove the protective film from the acrylic pieces, and mount the LCD module to the underside of the top panel (which has four holes to suit the LCD module) using the 25mm machine screws and M3 nuts.

Thread the tapped spacers onto the end of the rightmost (furthest away from the I^{2}C breakout board) machine screws by about 4mm.





Attach the 15mm nylon machine screws to the bottom panel in the middle holes using one nylon washer and nut on each. This acts as a spacer.

Attach the Uno to the machine screws using the remaining two nylon nuts.

Press the LC Meter shield onto the Arduino. Slot the end piece with the holes onto the Arduino, and drop it into the slots in the bottom panel, then fit the other three side panels so that the sides are all resting in the base.

Set the top onto the sides, ensuring the LCD headers push into the headers on the LC shield.

The halves are secured by threading the 32mm machine screws into the tapped headers to sandwich the assembly together.

We recommend adding self-adhesive rubber feet to protect your desk from the exposed screw heads at the bottom of the enclosure.

**Note that while the case here has been photographed grey (for clarity), it is actually crystal clear, as seen at top of page.*

Using it

On startup, the Uno performs calibration tests, storing the F1 reference oscillator frequency (just C1) and the F2 calibration frequency (C2 in parallel with C1). During this time, the six calibration constants which are stored in EEPROM are displayed as they are loaded.

Initially, instead of waiting a full second to count the number of cycles out of the oscillator, the *LC Meter* only counts for 100ms to save time. This first measurement is used to detect whether an inductor or capacitor is connected and whether its value falls in the range best measured by the time-constant method or the oscillator method.

If the frequency appears to be stable for two consecutive readings, the *LC Meter* performs another test each

Measuring the value of large inductors and capacitors

If we connect a large inductor or capacitor to our test oscillator, it will fail to oscillate. Hence, we need an alternative method to handle these sorts of components to make the LC Meter truly useful. The simplest method is charge-time measurement and we use digital output pins A0-A2, two fixed resistors (130Ω and 1.3kΩ) and analogue input pin A3 to perform this function.

If we know the value of the resistor and capacitor in a series RC circuit, we can calculate its 'time constant'. This is simply the product of the resistance in ohms and the capacitance in farads, giving a value with units of seconds.

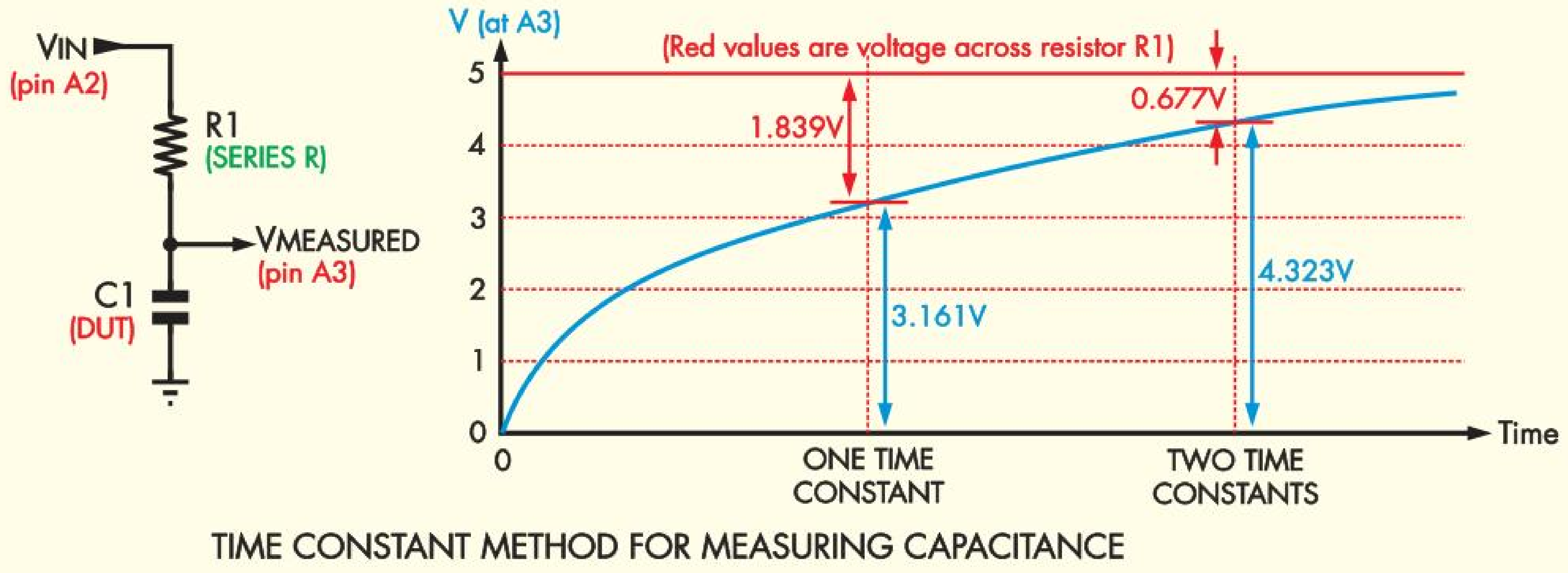


Fig.5: when testing high-value capacitors, C1 is charged via R1 and the voltage at pin A3 follows a curve similar to the blue one shown here. By measuring the voltage twice, and the time interval between measurements, we can determine its capacitance.

Refer to Fig.5. This shows the resistor and capacitor connected in the classic low-pass filter type arrangement. Assuming the capacitor starts fully discharged, the full applied voltage (VIN) appears across the resistor. After a period of one time constant, the voltage across the resistor will have fallen to VIN/e, where e is Euler's constant (2.718...).

In fact, regardless of what the voltage across the resistor is at the start, the ratio of the voltage at the start of the time constant period to the voltage at the end will be e.

This means we don't necessarily have to have the capacitor fully discharged to make our measurement, although we do get better accuracy if we work near this end of the curve, given the ~5mV resolution of the 10-bit ADC in the ATmega328 micro ($5V \div 2^{10}$).

To explain further, measuring over the first time constant period, the voltage across the resistor will drop from a nominal 5.000V down to 1.839V, a change of 3.161V or 647 ADC steps. Over the second time constant period, the voltage will change from 1.839V to 0.677V, a change of 1.163V or 238 ADC steps. Hence, starting with a mostly discharged capacitor gives us better measurement resolution.

We don't even need to measure an exact multiple of the time constant to complete the calculations, as we can use a logarithmic function to convert a ratio of voltages into a corresponding number of time constants.

For example, if we measure the voltage across the resistor at two different times and compute the ratio between the two of 4.95, this is equivalent to $e^{1.6}$ and that tells us that the period between the two measurements is equal to 1.6 time constants. So if the period happens to be 80ms, we can calculate the time constant is 50ms.

And if we know the value of the resistance, then we can easily compute the capacitance. In this case, if the resistance is 1.3kΩ, since t = RC, C = t / R or 38.5µF.

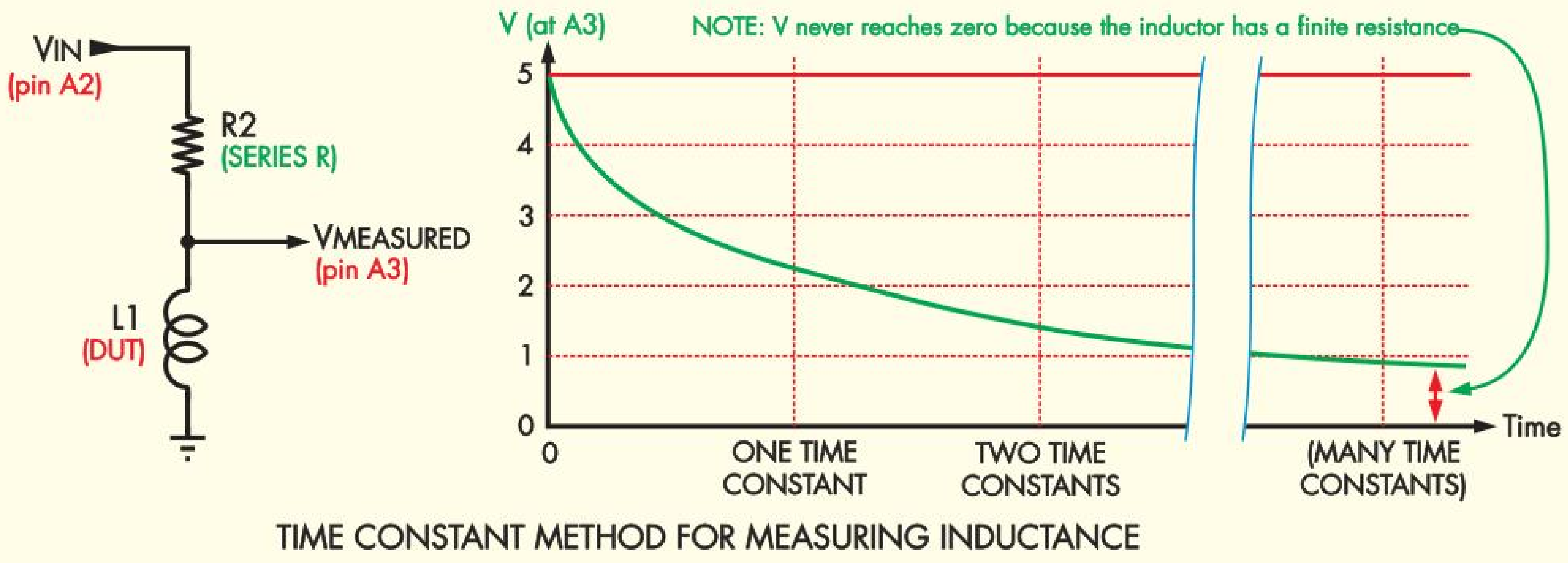


Fig.6: when testing high-value inductors, the voltage at A3 initially starts out at 5V and drops to a value determined by its series resistance once L1's magnetic field is fully charged. By measuring the voltage twice, and the time interval between measurements, we can determine its inductance.

A similar method for measuring the value of an inductor is shown in Fig.6. In this case, the curve is inverted and the equation for the time constant is the inductance in henries divided by the resistance in ohms (L/R).

Unlike capacitors, the ohmic resistance of a typical inductor can be quite significant and has to be taken into account in the calculation. With a capacitor, the voltage across the resistor will eventually get very close to 0V, only falling short due to leakage current, which is normally quite small. But with the inductor, in the steady state, the current is at a maximum and the voltage across it can be quite large.

Even so, the method is mostly the same; our starting state is with the inductor short-circuited to ensure no current is flowing, then the series resistor is connected to 5V and the time-voltage measurements begin.

After the time period has been measured, the series resistor is used to help determine the ESR of the inductor, and this is added to the series resistor's value before the time constant calculations are made, for improved accuracy.

Practical measurement ranges

For capacitors above 1µF with a series resistor around 1kΩ, the time constant is around 1ms, which is well within the realm of what an Uno can measure with reasonable accuracy. This design has no theoretical upper limit to what capacitance it can measure, given enough time. In practice, we had no trouble reading capacitor values up to 100,000µF. (That's as big as we had!).

We also tried measuring really high-value inductors (in the henries). The oscillator-based method actually works with many of these but its operation does malfunction sometimes. Our time-constant method consistently gave results within the spread measured by an expensive commercial LC meter.

Note that the same commercial LC meter, using different test frequencies, gave wildly different values for inductance, varying by 10% or more. So it seems that the values of some components can vary quite dramatically depending on the frequency that they are tested at.

Keep that in mind when using this LC Meter as the oscillator-based measurements are not at a fixed frequency.

cycle, this time taking one second for improved accuracy.

As long as the component remains connected, an average value is displayed by accumulating the results and dividing by the number of samples recorded. In this way, a highly accurate reading can be made.

The *LC Meter* uses the first two lines to display its initial estimate for inductance and capacitance.

It displays both, as we found there were a small number of cases (with very large capacitance and inductance values) where the *LC Meter* would detect one type of component as the other.

The measured oscillator frequencies are displayed to assist the user in following the operation of the *LC Meter*. The third line shows if the component is a capacitor or inductor, while the fourth line shows the averaged values and number of samples taken.

To use, simply connect the component across the leads and allow the reading to stabilise. Check that the component has been properly identified in the third line, and if you need an accurate value, allow a few readings to be recorded and read the average displayed on the fourth line.

Manual calibration

To access the calibration constants, you will need to connect to the serial port using a terminal program such as the Arduino IDE's serial monitor. The baud rate is 115,200 baud, with the standard Arduino defaults of 8 bits, no parity and one stop bit.

After opening the terminal program send a 'C' (capital) to enter calibration mode, pressing Enter if necessary to trigger sending the line of data. The *LC Meter* may take a second or two to respond, as it only checks the serial port once every test loop. The following menu appears:

```
Calibration Mode:
A:Enter R12 value – 130Ω
B:Enter R11 value – 1.3kΩ
C:Enter C2 value
D:Enter L1 value
E:Enter Cparasitic value
F:Enter Lparasitic value
G:Auto detect Cparasitic
  (leave terminals open circuit)
H:Auto detect Lparasitic
  (short circuit leads)
L:Load defaults
P:Print current values
S:Save to EEPROM
X:Exit calibration
Choose an option
```

So the procedure is select one of the 12 options from A to X.

Options A-F correspond to each of the six calibration constants. For A-D, the best way to improve the calibration is to measure the value of the component with an accurate meter and enter it. For example, to change the value of C2 to 1.1nF, type 'C' (and Enter if necessary). You will be prompted:

```
C selected.
Enter a value:
```

Type a value, including any of the SI multipliers from p (pico) to G (giga). For 1.1nF, we simply type '1.1n', with the units being assumed. If the wrong units are included, or you enter a negative number, an error message will appear. Otherwise, you will see:

```
0.0000000011000F
C2 changed to 1.100000nF
```

The *LC Meter* displays the entered value both with and without SI multipliers for clarity. At this point, the value is loaded into the program and will be used for measurements but it will not be saved in EEPROM for later use. To save the changes, select option 'S'.

Options 'G' and 'H' can be used to automatically detect stray capacitance and inductance.

When using option 'G' , ensure that no components are attached, although it is fine to leave leads attached if they are normally used for measuring – they will contribute to stray capacitance, so this can be accounted for in this calibration. If you have leads, make sure they are open-circuit.

Option 'G' simply runs a capacitance measurement on the leads (if connected), then saves this value to null future measurements.

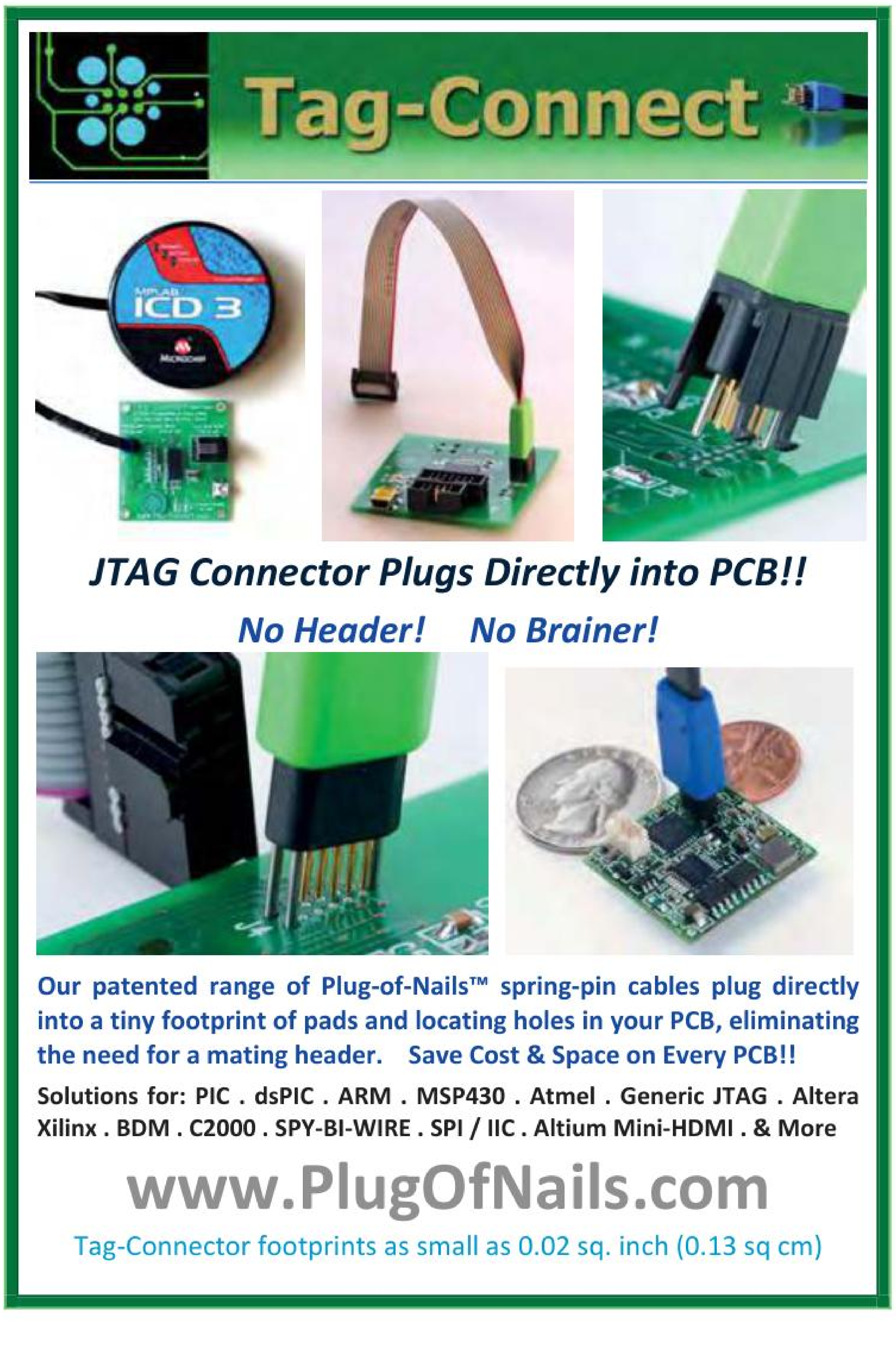
Option 'H' works in a similar fashion, although a shorting bar will need to be installed to create the effect of a zero inductance.

If you are using leads to measure in inductance mode, they can be shorted before selecting this option. Again, the value is saved and used to null inductance measurements. As with other values, these will need to be saved with option 'S'.

Option 'L' loads sensible default values (but does not save them to EEPROM) and gets you back to the initial condition.

Option 'P' displays the current values in use, while 'X' returns to measurement mode.

Notice that the LCD also echoes what is occurring on the serial monitor.



JTAG Connector Plugs Directly into PCB!!
No Header! No Brainer!
Our patented range of Plug-of-Nails™ spring-pin cables plug directly into a tiny footprint of pads and locating holes in your PCB, eliminating the need for a mating header. Save Cost & Space on Every PCB!!
Solutions for: PIC . dsPIC . ARM . MSP430 . Atmel . Generic JTAG . Altera Xilinx . BDM . C2000 . SPY-BI-WIRE . SPI / IIC . Altium Mini-HDMI . & More
www.PlugOfNails.com
Tag-Connector footprints as small as 0.02 sq. inch (0.13 sq cm)

Laser
Why tolerate when you can automate?
X-10 Home Automation
KAT5 AV transmission and IR control system
Barix Ethernet based MP3, communications and control systems
www.laser.com
Integrators, Installers, Trade and Retail customers welcome
C-Bus Shop
C-Bus and
C-Bus Wireless
www.cbus-shop.com
Laser Business Systems Ltd
Tel: +44 (0) 20 8441 9788
Fax: +44 (0) 20 8449 0430
Email: info@laser.com
16 Garthland Drive, EN5 3BB

Win a LAN7430 PCIe 3.1 Ethernet Bridge Evaluation Board

Practical Electronics is offering its readers the chance to win a Microchip LAN7430 PCIe 3.1 Ethernet Bridge Evaluation Board (EVB-LAN7430). The LAN7430 is a PCIe 3.1 (at 2.5GT/s) to Gigabit Ethernet bridge, providing an ultra-high-performance and cost-effective PCIe-to-Ethernet connectivity solution for embedded and automotive applications.

LAN7430 contains an integrated Ethernet PHY, PCIe PHY, PCIe endpoint controller, Ethernet MAC, Integrated OTP, JTAG TAP and EEPROM controller. The device supports 10BASE-T, 100BASE-TX and 1000BASE-T Ethernet with auto-negotiation, auto-polarity correction, HP Auto-MDIX support and is compliant with IEEE 802.3/802.3u/802.3ab standards.

PCIe-based networking provides flexibility for the routing and placement of network connections anywhere in the system. Specific enhancements for PCIe LPSS (low-power sub-states) L1.1 and LPSS L1.2 have been integrated to lower system power consumption.

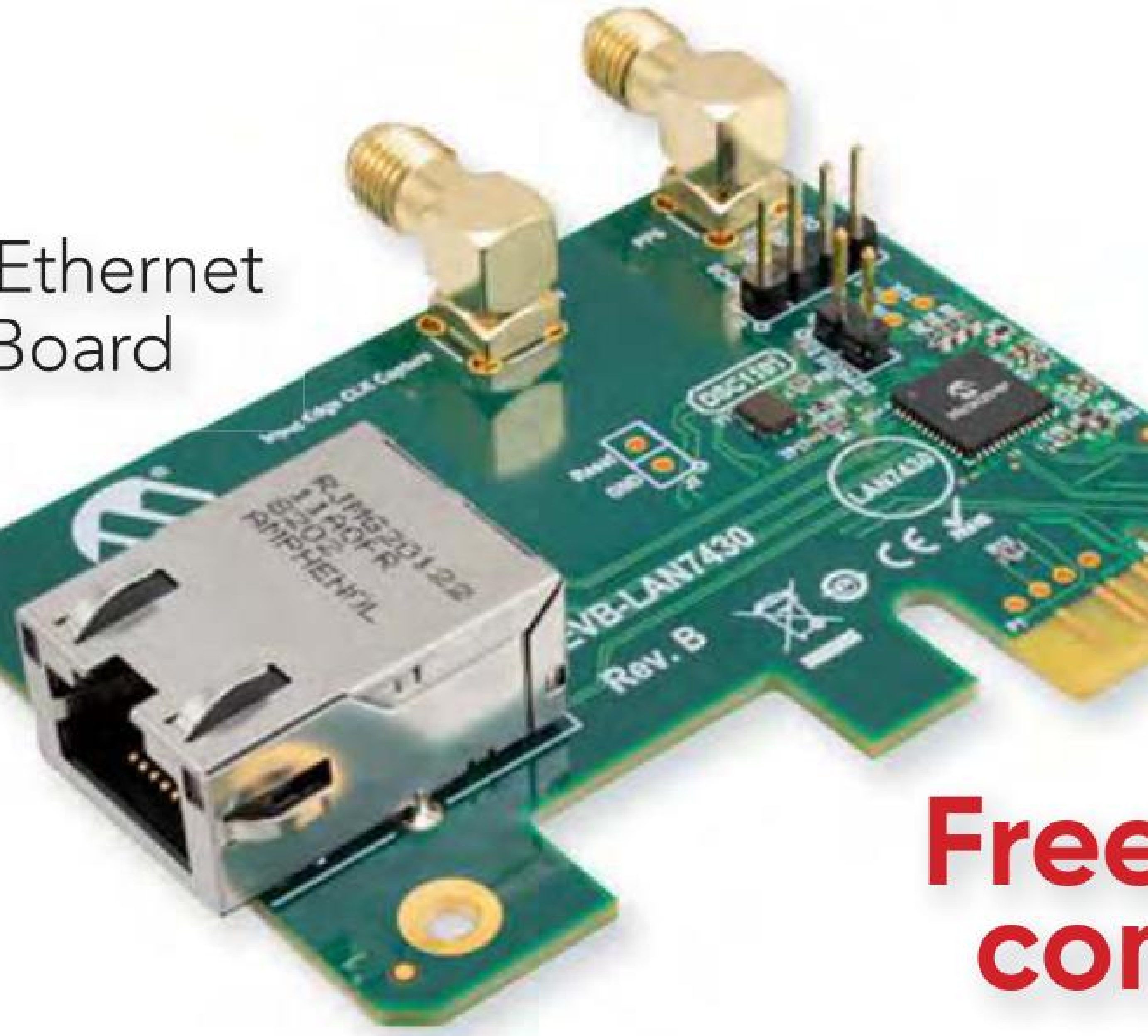
LAN7430 is available with a wide range of drivers including Windows, Linux, and Pseudo Code for development towards RTOS. LAN7430 also supports Precision Time Protocol using industry standard 1588-2008 PTP. GPIOs are provided that can be software defined for this function. Included within this feature is a programmable timer compare output that can also be software configured.

The LAN7430 integrates all needed voltage regulators and requires only a single 3.3V supply. Further BOM savings can be realised through the integrated clock tree, only requiring one external crystal or clock source. An EEPROM interface is provided, although it is not mandatory – an integrated OTP memory can provide for configuration data, further reducing system design cost and burden.

The LAN7430 is backed by Microchip's free LANCheck online design review service that reviews customer designs, ensuring that best design practices are used.

Worth $60.00 (approx £45.50) each

Microchip
LAN7430 PCIe 3.1 Ethernet Bridge Evaluation Board



Free-to-enter competition

How to enter

For your chance to win a LAN7430 PCIe 3.1 Ethernet Bridge Evaluation Board, enter your details in the online entry form at:

http://page.microchip.com/EPE-LAN7430.html

Closing date

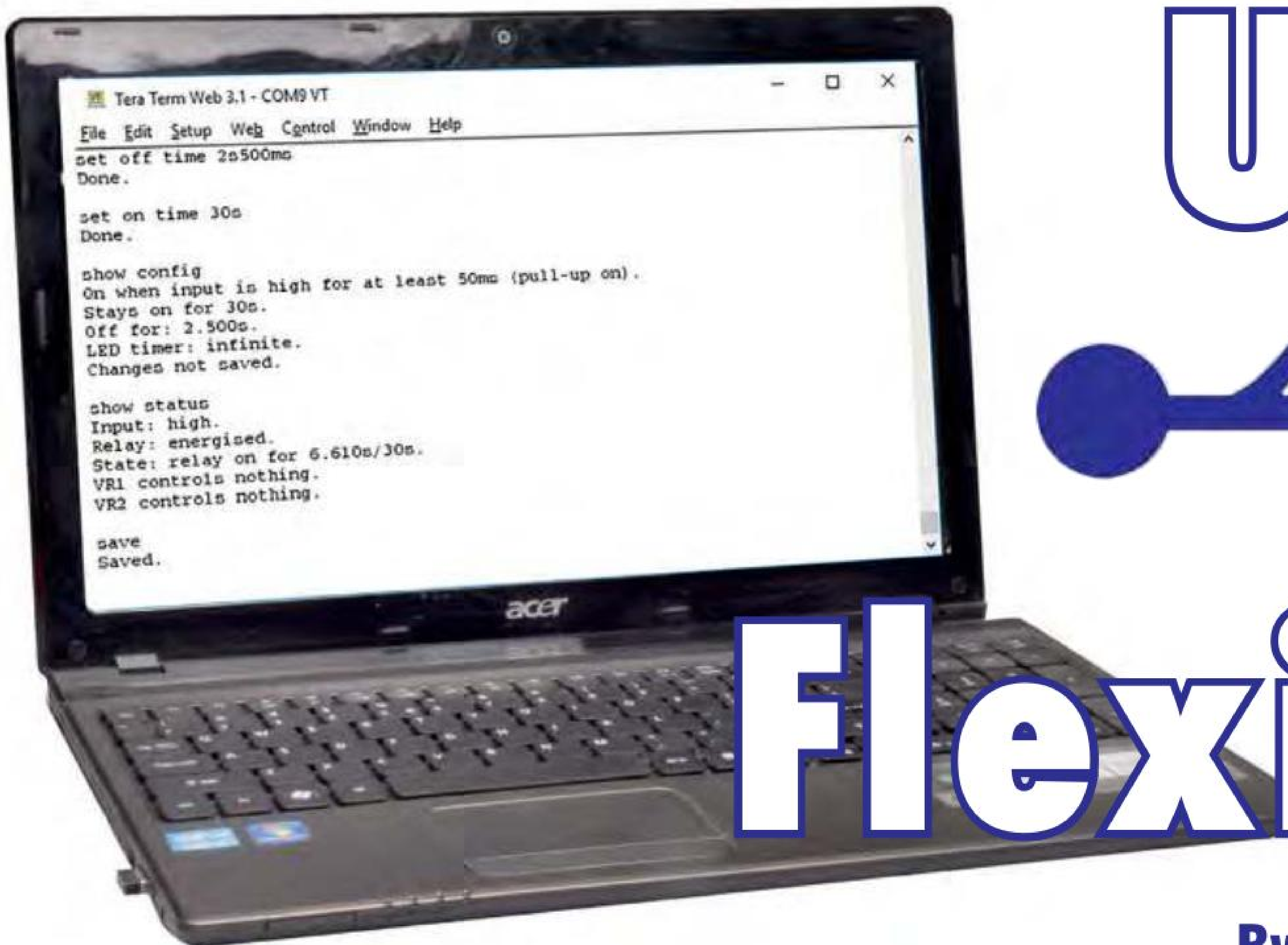
The closing date for this offer is 30 June 2019.

February 2019 winners

Stewart McSporran

He won an ATmega4809 Xplained Pro (ATmega4809-XPRO) worth $37.50

USB Flexitimer


Tera Term Web 3.1 - COM9 VT
File Edit Setup Web Control Window Help
acer

```
set off time 2s500ms
Done.

set on time 30s
Done.

show config
On when input is high for at least 50ms (pull-up on).
Stays on for 30s.
Off for: 2.500s.
LED timer: infinite.
Changes not saved.

show status
Input: high.
Relay: energised.
State: relay on for 6.610s/30s.
VR1 controls nothing.
VR2 controls nothing.

save
Saved.
```

By John Clarke and Nicholas Vinen

This very flexible timer can switch its relay at intervals from milliseconds up to many days. It can be triggered by an external pulse or set to run continuously, switching its relay on and off at a particular interval to control an external device. It's easy to build and you can set it up with two on-board trimpots or configure it with your laptop PC via its USB port.

We have published a number of *Flexitimers* over the years, the last one being a PIC-based unit described in August 2010. It was programmed with an array of jumper links which gave a lot of flexibility, but it required some mental agility to get the timing intervals you wanted and it has now been discontinued.

Our new *USB Flexitimer* also uses a PIC microcontroller, but you can program its various time intervals with two on-board trimpots or with your laptop PC via the USB port. That makes it easy to set precise intervals and allows those intervals to vary over a very wide range, from milliseconds to days.

Timers like this one are popular because they can be used in so many different situations.

For example, they can be used to switch on a light or a fan for a fixed period with the single press of a button. Or it can be used to switch power on to a device periodically; eg, open a solenoid valve for a minute every hour or power up a radio transmitter for a few minutes now and then.

It's also suitable for automotive uses; for example, as a turbo timer, to keep the engine running for a few minutes after you switch the ignition off, allowing the turbo to cool down. In fact, we won't even try to think of all the different uses you could put it to because there are simply too many to list.

It has an onboard DPDT relay, with 5A-rated contacts, which is switched on and off at the programmed intervals. It can run in a continuous loop, switching on and off at pre-defined intervals, or it can be triggered by an external switch, relay or digital signal.

There are various options to control how long the relay remains on when it's triggered externally. These are detailed below.

The two trimpots are optional, and if fitted, you can use them to adjust the time periods without needing a computer. The functions and ranges of both potentiometers are configurable via the USB interface. You can leave them off the board if you prefer to use the USB interface to change the timings.

While we've tried to make the unit easy to set up, we also wanted to make it truly flexible so that it can be used in the widest possible range of applications.

So it has quite a few different options which should allow you to set it up to suit virtually any circumstance. But you don't have to use them all; you can opt to keep the settings simple.

Features and specifications

- **Function: monostable or astable timer controlling a DPDT relay**
- **Timing period: on and off times programmable from 100ms to 50 days**
- **Timing adjustment: programmed over USB or using two trimpots (optional)**
- **Timing accuracy: typically ±0.25%**
- **Relay contact ratings: 30V DC/AC at up to 5A (8A with the Altronics relay)**
- **Trigger input: digital (<1V low, >1.6V high) with pull-down or pull-up**
- **Trigger modes: trigger on high level, low level, rising edge, falling edge or state change with option to reset timer on re-trigger**
- **Power supply: 12-15V DC; 2mA with relay off, 45mA with relay on (or USB 5V for setup and testing)**
- **Indicator LEDs: four LEDs, including power and relay state; two have adjustable time-outs**
- **Trigger delay: optional, can be set with 1ms resolution, min/max option**

This is actually the fifth iteration of the *Flexitimer* – the first was published in the March 1991 issue of *Electronics Australia*, and the most recent (*Flexitimer Mk.4*) as noted above, apeared in the August 2010 issue of *EPE*.

This one is better in just about every way, having a wider range of time settings, better accuracy, easier set-up and much more flexible options.

It uses a similar number of components and is around the same size. The power consumption of the new unit is lower, especially when the relay is not energised, and it includes extra indicator LEDs.

Circuit description

As shown in Fig.1, the circuit is based around PIC16F1455 microcontroller IC1, which has a built-in USB interface for programming it. It also has an internal analogue-to-digital converter (ADC) to sense the position of the optional trimpots (VR1 and VR2) which can be used to adjust the timings.

The USB D+ and D– data pins (pins 13 and 12) are connected directly to the USB socket, CON4. The USB +5V rail is connected to the micro's 5V supply rail via a 1N4148 diode (D3), allowing the unit to be powered by the connected computer for set-up and testing. D3 prevents current being fed back into the USB port if power is simultaneously applied to CON1.

The coil of DPDT relay RLY1 is driven by NPN transistor Q2, which in turn is driven by the micro's RC0 digital output (pin 10) via a 1kΩ current-limiting resistor. When pin 10 goes high, the relay is energised and the pairs of COM and NO terminals on CON3 are connected together.

When pin 10 is low, the relay is de-energised and the COM and NC terminals are connected instead.

When the relay is de-energised, the collapsing magnetic field in its coil causes a voltage to be induced across it and this is fed back into the 11.4V supply rail by diode D2, protecting Q2. LED4 is connected in parallel with the relay coil, along with a 10kΩ resistor and indicates when it is energised.

The trigger input at CON2 drives the base of NPN transistor Q1 via a 2:1 voltage divider comprising two 10kΩ resistors. The first 10kΩ resistor limits the base current of Q1 to a safe level while the second one acts as a pull-down, keeping it switched off if no voltage is applied at CON2.

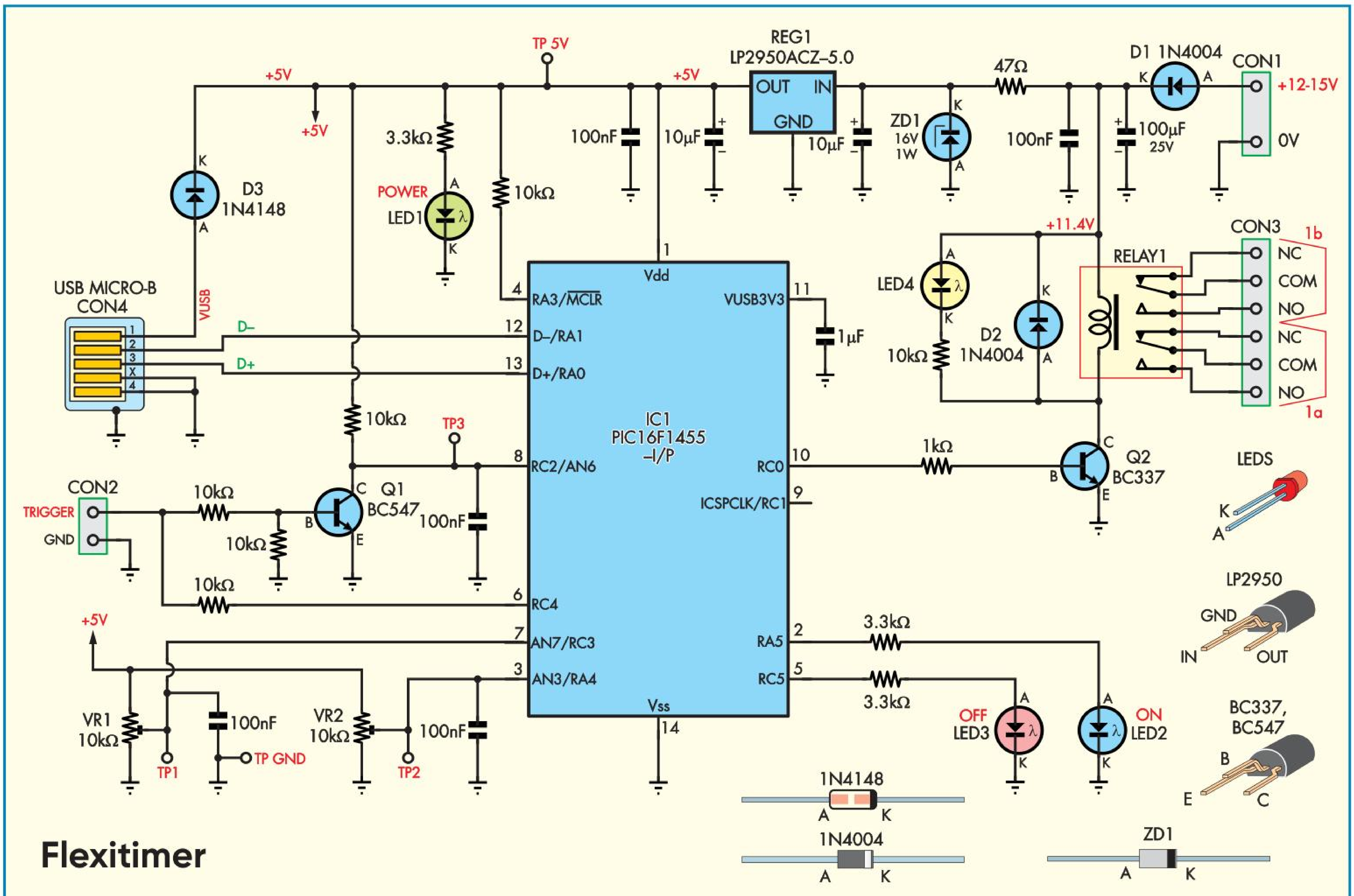
Q1's collector is pulled up to the +5V rail via a 10kΩ resistor, so it operates as an inverter – when the voltage at the trigger input is above 1.6V, input pin RC2 (pin 8) of IC1 is pulled low and when the trigger input is below 1.0V, pin 8 is high. The software compensates for this inversion.

A 100nF capacitor from pin 8 to ground prevents any brief spikes or glitches at the trigger input from being detected as a trigger event. It has a time constant of around 1ms in combination with the 10kΩ pull-up resistor. A further trigger delay can be configured in the software, as described below.

The 10kΩ resistor from pin 6 (digital output RC4) to the trigger input at CON2 gives the option of pulling the trigger input high, to +5V when it is not driven. This allows you to connect a relay or switch across CON2 and when it closes, it will pull the input low.

If the input is being actively driven high/low then it will trigger the unit regardless of the state of output pin 6, but setting it low will save a little bit of power. On the other hand, if you have a trigger source which actively pulls the input high but does not actively pull it low, you would need to switch the pull-up off for it to work properly. That can be done via the USB interface.

Trimpots VR1 and VR2 are connected across the 5V supply rail so that their wipers sweep over a 0-5V



Flexitimer
REG1
LP2950ACZ–5.0
OUT
IN
GND
TP 5V
+5V
D1 1N4004
CON1
+12-15V
0V
47Ω
100nF
10µF
10µF
ZD1
16V
1W
100nF
100µF
25V
3.3kΩ
POWER
LED1
10kΩ
D3
1N4148
USB MICRO-B
CON4
VUSB
D–
D+
+11.4V
CON3
1b
NC
COM
NO
NC
COM
NO
1a
RELAY1
LED4
10kΩ
D2
1N4004
Vdd
RA3/MCLR
D–/RA1
D+/RA0
VUSB3V3
1µF
IC1
PIC16F1455
–I/P
RC2/AN6
RC0
ICSPCLK/RC1
RC4
AN7/RC3
AN3/RA4
RA5
RC5
Vss
TP3
10kΩ
CON2
TRIGGER
GND
10kΩ
10kΩ
Q1
BC547
100nF
10kΩ
1kΩ
Q2
BC337
LEDS
+5V
VR1
10kΩ
100nF
TP1
TP GND
VR2
10kΩ
TP2
100nF
3.3kΩ
3.3kΩ
OFF
LED3
ON
LED2
LP2950
GND
IN
OUT
BC337,
BC547
B
E
C
1N4148
1N4004
ZD1
A
K

Fig.1: the heart of the *USB Flexitimer* is microcontroller IC1, which is programmed by a computer connected using USB port CON4 or using trimpots VR1 and VR2. Transistor Q1 provides signal conditioning for the trigger input while transistor Q2 drives the coil of RLY1.

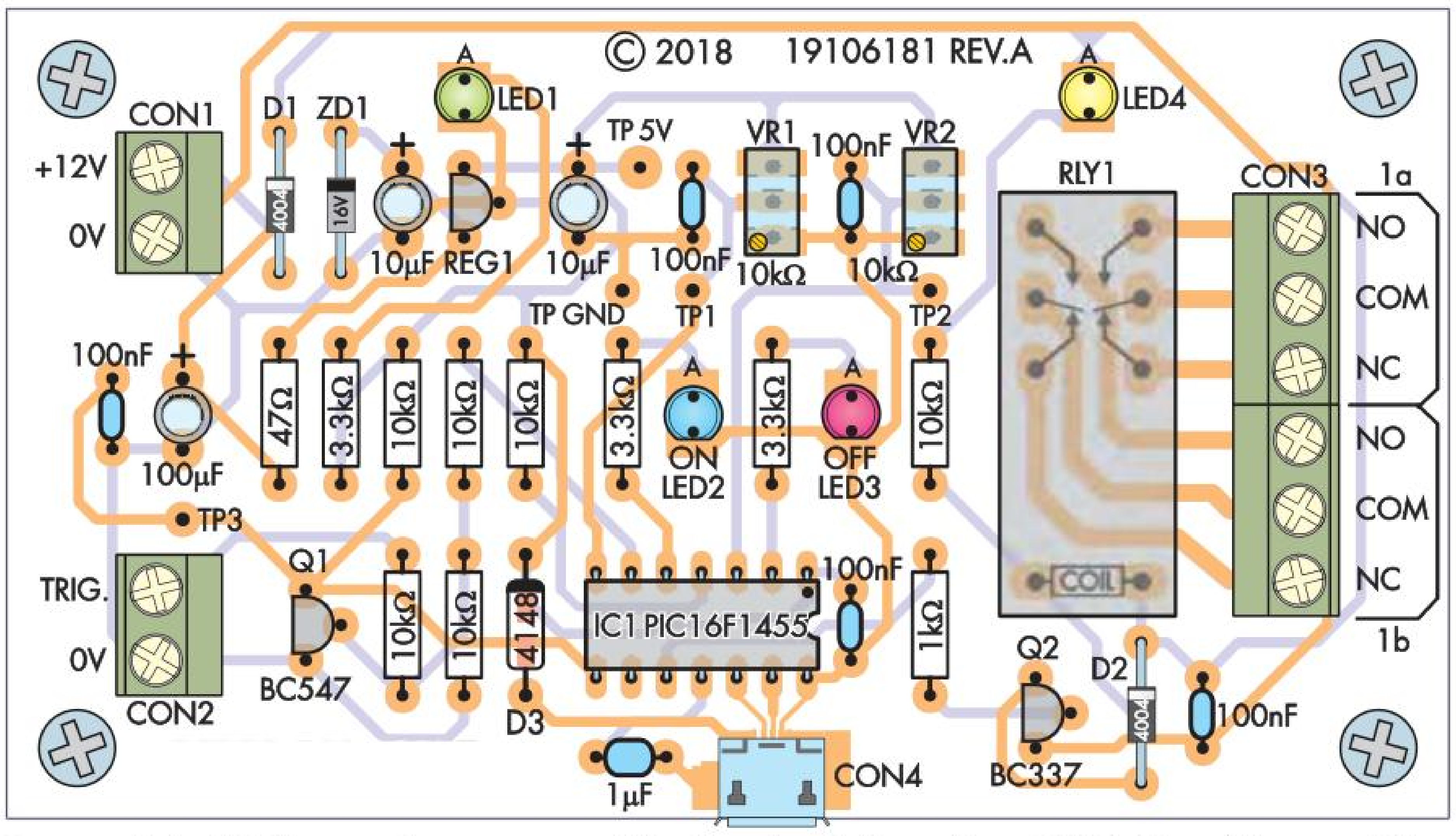
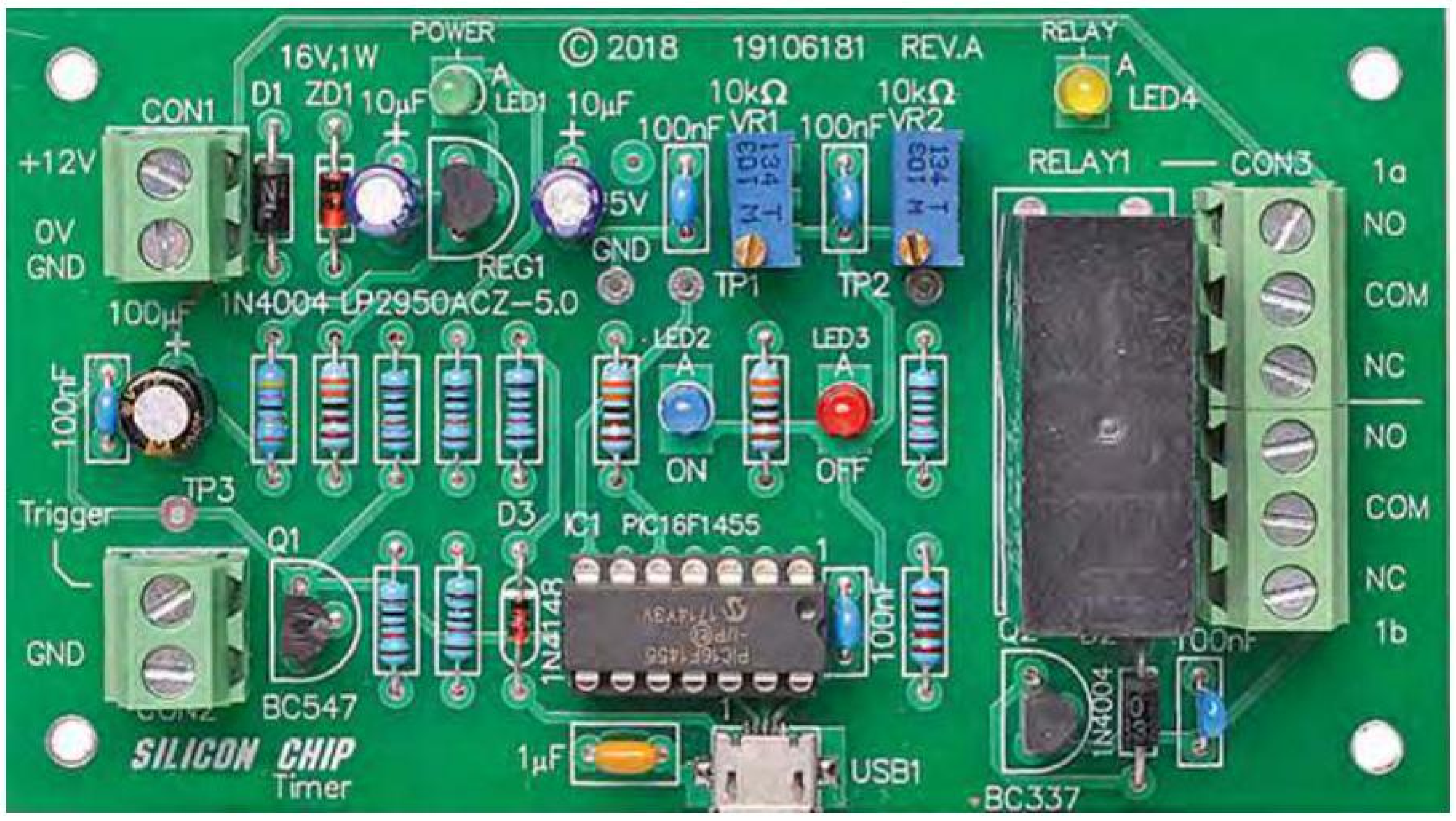


Fig.2: use this PCB overlay as a guide for building the *USB Flexitimer*. The USB socket CON4 is the only surface-mount device. IC1, D1-D4, VR1, VR2 and the electrolytic capacitors are polarised and must be installed as shown here. It can be fitted into a USB plastic Jiffy box or any other suitably sized enclosure.



This same-size photograph of the assembled PCB, in conjunction with the component overlay above, shows where each of the components goes.

Parts list USB Flexitimer

1 double-sided PCB, 104 x 58.5mm, coded 19106181 [*PE PCB Service*]
1 DPDT 12V DC coil relay (RLY1) [Jaycar SY-4052 (5A), Altronics S 4270A (8A)]
2 2-way terminal blocks, 5.08mm pin spacing (CON1,CON2)
2 3-way terminal blocks, 5.08mm pin spacing (CON3)
1 SMD micro USB socket (CON4)
1 14-pin DIL socket (optional, for IC1)
1 UB3 Jiffy box (optional)
4 short M3 tapped spacers and 8 short M3 machine screws (optional, for mounting in box)
2 cable glands (optional, for wiring when mounted in box)

Semiconductors

1 PIC16F1455-I/P microcontroller programmed with 1910618A.hex (IC1)
1 LP2950ACZ-5.0 micropower linear 5V regulator (REG1)
1 BC547 NPN transistor (Q1)
1 BC337 NPN transistor (Q2)
1 3mm green high-brightness LED (LED1)
1 3mm blue LED (LED2)
1 3mm red high-brightness LED (LED3)
1 3mm yellow high-brightness LED (LED4)
1 16V 1W zener diode (ZD1)
2 1N4004 1A diodes (D1,D2)
1 1N4148 small-signal diode (D3)

Capacitors

1 100µF 25V PC electrolytic
2 10µF 16V PC electrolytic
1 1µF 50V multi-layer ceramic or 63/100V MKT
5 100nF 63/100V MKT

Resistors (all 0.25W, 1% metal film)

6 10kΩ 3 3.3kΩ 1 1kΩ 1 47Ω
2 10kΩ 25-turn vertical trimpots (VR1,VR2)

range and they can be monitored at TP1 and TP2. These voltages are filtered by a 100nF capacitors (to keep the source impedance low, for maximum ADC accuracy) and then applied to analogue inputs AN7/AN3 of IC1 (pins 7 and 3).

By default, trimpot VR1 adjusts the on time over a range of 1-60 seconds and VR2 controls the off time over a range from zero to 60 seconds.

You can use the USB interface to change this. You can set the minimum and maximum time for each trimpot and you can also set them to control different parameters, such as the trigger delay, rather than the on and off times. (The various options will be described later.)

The power supply is quite simple but designed to be rugged for vehicular use. The 12-15V DC supply is applied to CON1 and passes through reverse-polarity-protection diode D1. The resulting 11.4V (nominal) supply is filtered by a 100µF capacitor and fed directly to the coil of relay RLY1.

This supply rail is then further filtered by a 47Ω series resistor and 10µF capacitor, and spikes are clamped by 16V zener diode ZD1. This filtered supply then feeds low-quiescent-current 5V regulator REG1, which supplies microcontroller IC1.

LED1, the power indicator, is connected across the 5V rail with a 3.3kΩ current-limiting resistor, giving an operating current of around 1mA.

There is a 10µF filter capacitor on the 5V rail and a 100nF bypass capacitor close to IC1, while IC1 also has a 10kΩ pull-up resistor for its $\overline{MCLR}$ reset input so that it is reset at power-up and then operates continuously.

LEDs 2 and 3 are driven from outputs RA5 (pin 2) and RC5 (pin 5) of IC1, with 3.3kΩ current-limiting resistors, giving them a similar brightness to LED1 and LED4. These can be set up so that they are lit for a limited time period after the relay switches, to save power, as described below.

Basic operation

Fundamentally, the *USB Flexitimer* operates as follows. Initially, it waits for the trigger condition to be met, eg, for the trigger input to go high or low. There are five different trigger options as described in the options panel.

There is also an optional trigger delay. If enabled, the input must remain stable in this condition for that period to trigger the timer.

Once triggered, the relay is energised and the timer runs for the specified on time. The relay is then de-energised and the timer runs for the specified off time.

Once the off time expires, it will go back into the initial state, waiting for a trigger event at the input.

By default, with nothing connected to CON2, the timer will run continuously, switching the relay on and off in a pattern. This is because (also by default) pin6 is held high and the trigger condition is a high level at CON2.

Thus the timer is automatically triggered at the start of the process.

Note that the relay is not energised during the off time and for the trigger delay time, so if you need a precise off-time and have a trigger delay time, you should subtract the delay time from the off time that you set.

Construction

The *USB Flexitimer* is built on a double-sided PCB, coded 19106181. It measures 104 × 58.5mm and is available from the *PE PCB Sevice*. This can be installed in a standard UB3 Jiffy box or any other enclosure with suitable internal dimensions.

Use the PCB overlay (Fig.2) and photo as a guide to fitting the components.

Pre-programmed microcontrollers are available from the SILICON CHIP Online Shop or you can program one yourself, if you have a suitable programming device, using the HEX file downloaded from the PE website.

Start by mounting the micro USB socket. Apply some flux paste to the five small pads and the two larger mounting pads. Place the part on the board, ensuring the two small plastic posts go into the corresponding holes in the PCB.

Check that the five small signal pins line up with their matching pads and then solder one of the side-mounting tabs to the board.

Now re-check the alignment of the signal pins. If it's off, re-melt that solder joint and nudge the part into it's correct position. Then solder the five signal pins.

It's very difficult to avoid bridging them, so it's easier to simply make sure all of them are soldered to the pads on the board and then apply some extra flux paste and use a piece of solder wick to remove the excess solder.

Next, solder the second mounting tab so that the socket is held rigidly in place.

You can then move on to the resistors. It's best to check their values with a DMM before soldering them in place. Follow with the diodes. There are three different types, so don't get them mixed up and make sure the cathode stripes are oriented as shown in Fig.2.

If you opt to use an IC socket, solder it in place now, ensuring that its notched end is oriented correctly. Otherwise, mount IC1 directly to the board with the same orientation.

Then install the 100nF and 1µF capacitors, followed by transistors Q1 and Q2 and regulator REG1. The small capacitors are not polarised.

You may need to bend the leads of the transistors out slightly to fit the PCB pad locations before fitting them. Do not get the transistors and regulators mixed up as they are in similar packages.

Next, fit terminal blocks CON1-CON3. CON3 consists of two 3-way terminal blocks which are dovetailed together before soldering. In each case, make sure that the wire entry holes face the nearest edge of the PCB.

If you are installing the optional trimpots (VR1 and VR2), do it now. Make sure you insert them into the PCB with the adjustment screw oriented as shown in Fig.2.

Now fit the electrolytic capacitors, ensuring that their longer lead (+) goes into the pads marked positive on the PCB. The striped side of the can indicates the negative lead.

You have the option of mounting LEDs1-4 right down on the PCB, as we have done on our prototype, or mounting them on longer leads so

Software details

You might think that the software for a timer would be simple. But due to the USB interface and the flexibility of this design, we were barely able to squeeze the required functions into the 14KB (8K words) of Flash memory available.

The firmware is written in the C language rather than assembly language for two reasons: one, it would have taken a lot longer to write in assembly language and two, the Microchip USB sample code is in C. It is converted to a HEX file using the Microchip XC8 compiler. The free version lacks full optimisations so you can't use it to compile this code as the result will not fit in the available Flash memory. (We supply a HEX file which the free software can load into the chip. But you need the paid-for compiler to re-generate the HEX file from the source code, which would be necessary if you wanted to make any changes to the way it works.)

The source code can be downloaded from the *Practical Electronics* website if you are interested in how it works.

To start with, we took the Microchip CDC (USB serial) sample program and stripped out all the parts we didn't need. This resulted in a program that simply echoed back what you type into the terminal and that code already took up around 30% of the available Flash memory.

Given the number of 'strings' (ie, sequences of characters) required for the command line interface in this project, we quickly realised that we would have to devise a custom method of storing these strings in Flash.

That's because the default behaviour of the XC8 compiler is to store each character in one Flash word (14 bits). So the ~1000 characters we needed would have taken up 1K words, or 12.5% of the available Flash.

We're not using accented characters, just basic ASCII, so all our characters have codes in the range of 0-127. That means they will fit in 7 bits and hence, each 14-bit flash word could be used to store two characters, not one.

So we wrote a little program that took a list of strings and packed it into 14-bit program words, then produced an ASM file to link in with the C file. We also needed two new routines, one to unpack the words and write the strings into a memory buffer (identified by their offset in the 'blob' [en.wikipedia.org/wiki/Binary_large_object]) and another to compare the packed words to the contents of a memory buffer, for parsing commands typed by the user.

Overall, this freed up about 5% of the Flash memory for extra code. And the program takes up over 99% of the Flash so we certainly needed that breathing room.

We also wrote our own functions to convert numbers to and from ASCII strings, also for the command line interface, as they turned out to be smaller than the equivalent C library functions (although we do make use of the 'strtoul' built-in function, to convert a string to a 32-bit integer).

Timing is based on the micro's internal oscillator, which runs at 12MHz and uses 'active clock tuning', to lock it to the USB host's clock so that it meets USB timing specifications. This is used to increment a 16-bit variable in memory at 1ms intervals using the internal TIMER1 hardware with a pre-scaler value of 16 and pre-loading the TMR1 register with the value 64786, which causes the timer 1 roll-over interrupt to trigger 750 intervals later. 750 x 16 = 12,000, giving us our 1ms interval relative to the 12MHz system clock.

This millisecond counter variable is then used by the logic in the main loop of the software to determine how long to wait and when to change the relay state based on the trigger input logic level. It also makes use of the configuration data, which is copied from the high-endurance area of Flash memory to RAM at start-up.

Much of the time spent developing this software consisted of finding ways to rewrite code to take up less Flash space. Many of these optimisations involved moving commonly used code sequences into functions to avoid them having to be stored in multiple locations in Flash. But this didn't always help since function call overhead on the PIC16 is fairly high.

In fact, we found the results of the XC8 compiler to be somewhat unpredictable; one seemingly inconsequential change could result in a 15% increase or decrease in Flash memory utilisation! So a lot of experimentation was required to find the best combination of code to achieve the desired result.

Configuration options

There are six settings that control the operation of the Flexitimer. They are: the trigger condition, four timing values (with some extra options) and whether the input pull-up resistor is enabled. Each is explained below. The four timing values are: the on time, the off time, the trigger delay time and the indicator LED on-time.

When reading the following explanation, keep in mind that the relay is energised while the main (on) timer is running and de-energised the rest of the time.

Trigger conditions

There are five possible input trigger conditions which will start the timer. These are when the input is a high level, low level, on a rising edge, on a falling edge or on either edge (input state change).

The high-level and rising-edge settings operate in a very similar manner, as do the low-level and falling-edge settings. The difference is what happens after the timer has elapsed. If you have the input trigger condition set to high level and the input is still high after the timer has elapsed, it will start again.

But if you have it set to rising edge, it will not. The input will have to go low and then high again to re-start the timer. It doesn't matter whether it goes low during the on time and then high later, or low after the timer has expired and then high again; what matters is the low-to-high transition must occur after the timing period has expired.

Similarly, in the input state change mode, the transition must occur after the timing period has finished to start a new one.

Having said all that, there is a separate option to reset the timer if a trigger condition occurs during the timing period. If this is enabled and the trigger conditions are met during the timer period, it will start over. That option is explained below.

The on time

This controls how long the relay is energised. While the resolution is 1ms, the minimum practical value is around 100ms, given that the relay will take around 10ms to switch on and off. The maximum time is 1193 hours, which works out to almost 50 days.

Like the other time options, the on-time is set using a string which looks like one of the following: '1h15m23.572s' or '1500ms' or '75m'.

If you specify an exact on time then each time the unit is triggered, the relay will be on for exactly that period. However, you can also specify it as a minimum or maximum on time.

If specified as a maximum, the timer will be cancelled and the relay de-energised if the input changes state before the timer has expired. So the timing period will range from a few milliseconds up to the period specified.

Conversely, if specified as a minimum, the relay will be energised for the time period specified and then, if the input has not changed state during that period, it will remain energised until the input does change state. So the timing period ranges from the time specified up to the length of the input trigger pulse.

Finally, there is the option to make the on time 'resettable'. This means that if the input trigger condition is met during the on time (including any delay, as described below), the timer starts again from zero. So if the input is repeatedly triggered, the on time will be extended each time. It does not make sense to specify this with a maximum on time, but it can be used in combination with an exact or minimum on time.

Off time

As soon as the timer expires or is cancelled, the relay is de-energised. You can set the off time to zero, in which case, as soon as the trigger conditions are met again, the timer will start again from zero and the relay will be re-energised.

However, if you specify a non-zero off time, then another timer is started and nothing will happen until it expires. You can use this to enforce a minimum time between the relay being de-energised and then re-energised.

It's also useful in a situation where you want the relay to be switched on and off at a particular interval or cadence. In this case, you can set the on time and the off time and arrange for the trigger condition to always be true. The relay will then continually switch on and off at the specified intervals.

As with the on time, the off time can range from 1ms up to nearly 50 days, but again, the minimum practical off time is around 100ms.

Trigger delay time

If the trigger delay time is set to zero then the timer will start and the relay will be energised as soon as the trigger conditions are met (either immediately after power-on or after the off time has elapsed).

However, you can specify a non-zero trigger delay time, in which case the input level will need to be stable for this period before the timer will actually start, and the relay will remain de-energised during this time. You can also use this as a way to purposefully insert a delay between the trigger signal and the relay being energised.

The range of the trigger delay time is the same as the other times; however, it would generally be a shorter period, from a few milliseconds to a few seconds.

It's also possible to have a fixed trigger delay period, which simply means that it doesn't matter what the trigger input does during the delay period; as long as it initially met the trigger conditions, after the fixed delay, the relay will be energised. This would be useful, for example, if you want to trigger the timer with the push of a momentary pushbutton but have a delay between that button press and the relay switching.

There's also a third option for the trigger delay and that is to specify it as a maximum time. In this case, once the trigger condition has been met, the unit waits for the input to change state and if the period of this trigger pulse is less than or equal to the specified maximum, the on timer starts and the relay is energised. Otherwise, the trigger pulse is ignored.

For example, if you were using a photo interrupter to trigger the unit, this would allow you to set it up to be triggered by fast-moving objects but not slow-moving ones.

LED on time

By default, LED2 lights whenever RLY1 is energised and LED3 lights whenever RLY1 is de-energised. However, you can specify a finite LED on time. In this case, LED2 lights as soon as RLY1 is energised and then switches off after the LED on time has elapsed. Similarly, LED3 lights as soon as RLY1 is de-energised, for the same time period.

If running the USB Flexitimer off a battery, you may want to minimise its power consumption, in which case you could set the LED on time to be quite short and leave LED1 off entirely. In this case, its standby current will be around 1mA without VR1 and VR2, or 2mA if they are fitted.

---

that they will protrude through the lid of the case.

Alternatively, you can chassis-mount some or all of them and wire them to the board with flying leads.

You can choose to leave LED1 and/or LED4 off if you need to minimise current consumption. Regardless of how you mount the LEDs, make sure that their longer (anode) leads connect to the pads marked 'A' on the PCB.

Now push the relay fully onto the PCB and solder its pins.

Finally, if you have fitted an IC socket for IC1, straighten its pins and insert it into the socket now, taking care that you don't fold any of the leads under the package when doing so.

Testing

The easiest way to test the completed board is to plug it into a USB port on your computer using a standard type A to micro type B cable.

Changing the configuration via USB

Plug the unit into your computer and connect the terminal emulator as per the testing instructions. You can now issue the following commands (shown here in `red`) to change the unit's configuration. Note that the changes will not be retained after power is switched off unless you issue a 'save' command. Also, you need to press Enter/Return after typing a command and check that you get a positive response.

help

Displays a short list of these commands.

pull-up (on|off)

Sets whether the input pull-up resistor is driven or not; eg:

```
pull-up on
Done.
```

set trigger on (high|low|rising|falling|change)

Issues this command to set the trigger condition to one of the five possibilities listed; eg:

```
set trigger on rising
Done.
```

set on time [min|max] [resettable] <time>

Sets the on time to a fixed value. The 'min', 'max' and 'resettable' keywords are optional and must be provided in that order (if using both). The time is specified as stated earlier*; eg:*

```
set on time min resettable
1h15m10.5s
Done.
```

set on time [min|max] [resettable] VR(1|2) <time> to <time>

Sets the on time to a variable value controlled by either VR1 or VR2, over the specified range; eg:

```
set on time max VR1 1m to 1h
Done.
```

set off time <time>

Sets the off time to a fixed value; eg:

```
set off time 30s
Done.
```

set off time VR(1|2) <time> to <time>

Sets the off time to a variable value controlled by either VR1 or VR2, over the specified range; eg:

```
set off time VR2 100ms to 1500ms
Done.
```

set trigger delay [min|max|fixed] <time>

Sets the trigger delay to a fixed value. If you don't specify 'min', 'max' or 'fixed', the default is 'min'; eg:

```
set trigger delay 25ms
Done.
```

set trigger delay [min|max|fixed] VR(1|2) <time> to <time>

Sets the trigger delay to a variable value controlled by either VR1 or VR2, over the specified range; eg:

```
set trigger delay fixed VR1
10ms to 100ms
Done.
```

set led timer <time>

Sets how long LED2/LED3 stay on once the relay state changes. Instead of a time value, you can specify 'infinite' so that they are continuously lit; eg:

```
set led timer 30s
Done.
```

set led timer VR(1|2) <time> to <time>

Sets the on time for LED2/LED3 to a variable value controlled by either VR1 or VR2, over the specified range; eg:

```
set led timer VR2 1s to 10s
Done.
```

show [active] config

Shows the unit's current configuration, ie, that which it is actively using to control the relay. Includes any changes you have made since power was applied, even if they haven't been saved yet; eg:

```
show config
On when input is high for at least 50ms (pull-up on).
Stays on for 1s to 1m (VR1).
Off for: 0ms to 1m (VR2).
LED timer: infinite.
```

show status

Displays the current input state, whether the relay is energised, the current timer value, the positions of the trimpots and the corresponding time values. Can be helpful as a debugging aid if the unit is not doing what you expect; eg:

```
show status
Input: high.
Relay: energised.
State: relay on for 31.226s/1m.
VR1 controls relay on-time (now 100% = 1m).
VR2 controls off-time (now 51% = 30.600s).
```

show saved config

Shows the unit's stored configuration; ie, that which is loaded at power-up. Does not include any unsaved changes you have made.

save

Saves any changes made to the configuration into the EEPROM. They will, therefore, be applied each time the unit is powered up.

revert

Discards any changes made to the configuration and loads the previous configuration from the EEPROM. Has the same effect as power cycling the device without saving the changes.

LED1 should light up immediately and your computer should identify a new USB device. Most operating systems (including Windows 10) should not need a driver as it appears as a standard serial device.

Note that the USB cable powers the PCB at this point. There is no need to connect a DC power supply. That step comes later.

In Windows 10, you can verify that it is working by opening Device Manager and expanding the 'Ports (COM and LPT)' section. It should appear as 'USB Serial Port (COMx)', where x is a number.

Next, you need to open a terminal program (Tera Term Pro is a good choice) and set it to use that serial port (in the Setup → Serial Port menu). The baud rate and other settings do not matter. Then return to the terminal emulator and type 'help' and then press Enter/Return.

You should see the help command echoed back to you as you type and a list of commands should then be displayed when you press Enter/Return. This verifies that the USB interface and microcontroller are working.

The default configuration results in the unit being self-triggered because the input pull-up resistor is active and the trigger condition is on a high input. As a result, you should see LED4 flashing at a rate determined by the positions of VR1 and VR2 (or at random intervals if those potentiometers have not been fitted). You should also see LED2 and LED3 switching on and off at the same time as LED4 changes state.

If you have fitted VR1 and VR2, adjust them and check that the on time and off time of LED4 vary as expected.

Otherwise, you can issue commands such as: 'set on time 1s' and 'set off time 1s' to change the on and off time – and then check that they do in fact vary as expected.

If you short the terminals of CON2 then LED4 should stay off once it switches off as the unit is no longer being triggered.

At this point, you could hook up a 12-15V DC supply to CON1 and check that the relay clicks on and off at the same time that LED4 lights up or goes dark.

Setting it up

Now that you have it connected to your PC, this is a good opportunity to set up the configuration to your requirements.

See the panel on the previous page, which describes changing the configuration for the list of commands that you can use to set it up. It's a good idea to start by issuing the `show config` command to see the current (default) settings. Don't forget to use the `save` command when you have finished.

You'll probably want to read through the panel on timing options first, to understand how the unit works, so that you can figure out how best to set it up for your particular application.

If you're setting it up to activate with an external trigger, you can simulate this by shorting the trigger pin on CON2 to either +12V or GND, to pull the input high or low.

The default configuration is as follows. The input trigger condition is on a high level with a 50ms (minimum) trigger delay.

VR1 varies the on time over the range of 1-60s, which is set in exact mode (not minimum or maximum). VR2 controls the off time over the range of 0-60s. LED3 and LED4 are constantly illuminated. The pull-up resistor is enabled and timer resetting by the trigger input is disabled.

If you're having trouble getting the timer to operate in the intended manner, you can plug it into the USB port of a computer and use the `show status` command to see what it is doing.

The result includes information on whether the device has been triggered, which timer is currently in operation, how long it has been running for and when the next state change will occur.

Housing it

The PCB is designed to fit inside a UB3 Jiffy box (130 × 67 × 44mm). It has a mounting hole in each corner so that it can be attached to the base of the box using M3 tapped spacers and short machine screws.

Alternatively, you can mount it inside some other piece of equipment, possibly that which it is switching on and off.

If you do decide to mount it in a UB3 Jiffy box, you could fit cable glands at either end for power wires and for the wires which connect to the *Flexitimer* relay contacts.

If you want to be able to re-program it while inside the box, you will also need to make a rectangular cut-out in the side to access the USB socket.

Reproduced by arrangement with
SILICON CHIP magazine 2019.
www.siliconchip.com.au

CRICKLEWOOD ELECTRONICS
Established 1981

Your best bet since MAPLIN
Chock-a-Block with Stock

Visit: www.cricklewoodelectronics.com
Or phone our friendly knowledgeable staff on **020 8452 0161**
Components • Audio • Video • Connectors • Cables
Arduino • Test Equipment etc, etc

Visit our Shop, Call or Buy online at:
www.cricklewoodelectronics.com
020 8452 0161
Visit our shop at:
40-42 Cricklewood Broadway
London NW2 3ET

ESR Electronic Components Ltd

All of our stock is RoHS compliant and CE approved. Visit our well stocked shop for all of your requirements or order on-line.

We can help and advise with your enquiry, from design to construction.

3D Printing • Cable • CCTV • Connectors • Components • Enclosures • Fans • Fuses • Hardware • Lamps • LED's • Leads • Loudspeakers • Panel Meters • PCB Production • Power Supplies • Relays • Resistors • Semiconductors • Soldering Irons • Switches • Test Equipment • Transformers and so much more…

Monday to Friday 08:30 - 17.00, Saturday 08:30 - 15:30

Station Road
Cullercoats
North Shields
Tyne & Wear
NE30 4PQ

Tel: 0191 2514363 sales@esr.co.uk www.esr.co.uk

ETI BUNDLE (1) Teach-In 3, 4 and 5 – all on CD-ROM – only £18.95

ELECTRONICS TEACH-IN 3 – CD-ROM

Mike & Richard Tooley

The three sections of the *Teach-In 3* CD-ROM cover a huge range of subjects that will interest everyone involved in electronics – from newcomers to the hobby and students to experienced constructors and professionals.

The first section (80 pages) is dedicated to *Circuit Surgery*, *EPE/PE's* regular clinic dealing with readers' queries on circuit design problems – from voltage regulation to using SPICE circuit simulation software.

The second section – *Practically Speaking* – covers hands-on aspects of electronics construction. Again, a whole range of subjects, from soldering to avoiding problems with static electricity and identifying components is covered. Finally, our collection of *Ingenuity Unlimited* circuits provides over 40 circuit designs submitted by readers.

The CD-ROM also contains the complete *Electronics Teach-In 1* book, which provides a broad-based introduction to electronics in PDF form, plus interactive quizzes to test your knowledge and TINA circuit simulation software (a limited version – plus a specially written TINA Tutorial).

The *Teach-In 1* series covers everything from electric current through to microprocessors and microcontrollers, and each part includes demonstration circuits to build on breadboards or to simulate on your PC.

ELECTRONICS TEACH-IN 4 – CD-ROM
A BROAD-BASED INTRODUCTION TO ELECTRONICS

Mike & Richard Tooley

The *Teach-In 4* CD-ROM covers three of the most important electronics units that are currently studied in many schools and colleges. These include, Edexcel BTEC level 2 awards and the electronics units of the Diploma in Engineering, Level 2.

The CD-ROM also contains the full *Modern Electronics Manual*, worth £29.95. **The Manual** contains over 800 pages of electronics theory, projects, data, assembly instructions and web links.

A package of exceptional value that will appeal to anyone interested in learning about electronics – hobbyists, students or professionals.

ELECTRONICS TEACH-IN 5 – CD-ROM
JUMP START

Mike & Richard Tooley

15 design and build circuit projects for newcomers or those following courses in school and colleges.

The projects are: ■ Moisture Detector ■ Quiz Machine ■ Battery Voltage Checker ■ Solar-Powered Charger ■ Versatile Theft Alarm ■ Spooky Circuits ■ Frost Alarm ■ Mini Christmas Lights ■ iPod Speaker ■ Logic Probe ■ DC Motor Controller ■ Egg Timer ■ Signal Injector Probe ■ Simple Radio Receiver ■ Temperature Alarm.

PLUS

PIC'n' Mix – starting out with the popular range of PIC microcontrollers and *Practically Speaking* – tips and techniques for project construction.

The CD-ROM also contains:

- ■ Complete *Teach-In 2* book, a practical introduction to PIC microprocessors
- ■ MikroElektronika, Microchip and L-Tek PoScope software.

Three Teach-ins for the great price of £18.95

PLUS you also get the contents of the free CD-ROM from each issue...

... so that's another TWO Teach-Ins and The Full Modern Electronics Manual!

What a Bargain!!

ETI BUNDLE (2) Teach-In 6, 7 and 8 – all on CD-ROM – only £18.95

ELECTRONICS TEACH-IN 6 – CD-ROM
A COMPREHENSIVE GUIDE TO RASPBERRY Pi

Mike & Richard Tooley

Teach-In 6 contains an exciting series of articles that provides a complete introduction to the Raspberry Pi, the low cost computer that has taken the education and computing world by storm.

This latest book in our *Teach-In* series will appeal to electronics enthusiasts and computer buffs who want to get to grips with the Raspberry Pi.

Teach-In 6 is for anyone searching for ideas to use their Pi, or who has an idea for a project but doesn't know how to turn it into reality. This book will prove invaluable for anyone fascinated by the revolutionary Pi. It covers:

- ■ Pi programming
- ■ Pi hardware
- ■ Pi communications
- ■ Pi Projects
- ■ Pi Class
- ■ Python Quickstart
- ■ Pi World
- ■ ...and much more!

The *Teach-In 6* CD-ROM also contains all the necessary software for the series, so that readers and circuit designers can get started quickly and easily with the projects and ideas covered.

ELECTRONICS TEACH-IN 7 – CD-ROM
DISCRETE LINEAR CIRCUIT DESIGN

Mike & Richard Tooley

Teach-In 7 is a complete introduction to the design of analogue electronic circuits. It is ideal for everyone interested in electronics as a hobby and for those studying technology at schools and colleges. The CD-ROM also contains all the circuit software for the course, plus demo CAD software for use with the *Teach-In* series.

- ■ Discrete Linear Circuit Design
- ■ Understand linear circuit design
- ■ Learn with 'TINA' – modern CAD software
- ■ Design simple, but elegant circuits
- ■ Five projects to build:
  - i) Pre-amp
  - ii) Headphone Amp
  - iii) Tone Control
  - iv) VU-meter
  - v) High Performance Audio Power Amp.

PLUS

Audio Out – an analogue expert's take on specialist circuits

Practically Speaking – the techniques of project building.

ELECTRONICS TEACH-IN 8 – CD-ROM
INTRODUCING THE ARDUINO

Mike & Richard Tooley

Hardware: learn about components and circuits
Programming: powerful integrated development system
Microcontrollers: understand control operations
Communications: connect to PCs and other Arduinos.

Teach-In 8 is an exciting series designed for electronics enthusiasts who want to get to grips with the inexpensive, popular Arduino microcontroller, as well as coding enthusiasts who want to explore hardware and interfacing. It will provide a one-stop source of ideas and practical information.

The Arduino offers a truly effective platform for developing a huge variety of projects; from operating a set of Christmas tree lights to remotely controlling a robotic vehicle through wireless or the Internet.

Teach-In 8 is based around a series of practical projects with plenty of information for customisation.

This book also includes *PIC n' Mix*: 'PICs and the PICkit 3 – A Beginners guide' by Mike O'Keefe and *Circuit Surgery* by Ian Bell – 'State Machines part 1 and 2'.

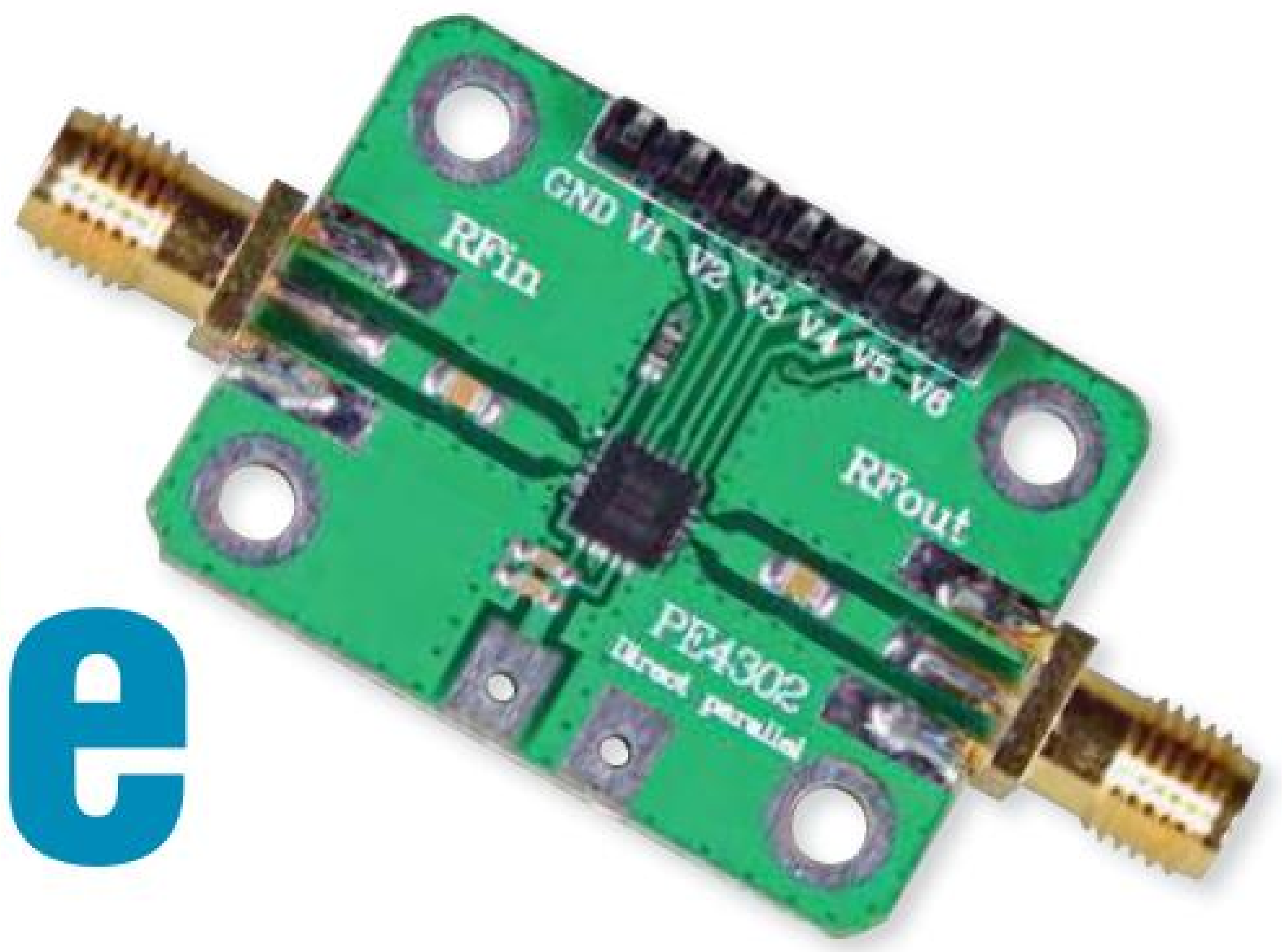
The CD-ROM includes the files for:

- ■ *Teach-In 8*
- ■ Microchip MPLAB IDE XC8 8-bit compiler
- ■ PICkit 3 User Guide
- ■ Lab-Nation Smartscope software.

ORDER YOUR BUNDLE TODAY!
JUST CALL 01202 880299 – OR VISIT www.epemag.com

Using Cheap Asian Electronic Modules Part 17: by Jim Rowe

4GHz Digital Attenuator Module



This highly linear 4GHz digital attenuator is programmable over 0dB to –31.5dB in 0.5dB steps. This type of attenuator is important in reducing signal levels to avoid overload in a mixer or amplifier. It could also be the basis of a precision full-range attenuator in an RF signal generator.

This digitally programmed step attenuator module (**banggood.com**, part ID 1167649), has six internal cascaded attenuators which can be switched in or out independently, to provide an overall attenuation range of 0dB to –31.5dB in 0.5dB steps. The operating frequency range of the module is from about 1MHz up to 4GHz.

The module's PCB is just 33 × 24.5mm in size. The RF input and output SMA connectors are edge-mounted on each end of the PCB; while the power and programming inputs are on the sides.

It uses a PE4302 IC in a 20-lead QFN (SMD) package measuring 4 × 4mm, from the San Diego-based (California) company Peregrine Semiconductor.

Their website (**www.psemi.com**) shows that they're part of the Murata group and they make a wide range of RF ICs based on their patented 'UltraCMOS' process, which is an advanced form of 'silicon on insulator' (SOI) technology. It now appears that the PE4302 is 'obsolete', having been replaced by the improved PE4312. However, it is still available, is much cheaper than the PE4312 and gives acceptable performance for non-critical applications. Banggood sell the module for around £6.75, but you can also purchase it from **eBay.co.uk** or AliExpress.

Fig.1 is the block diagram of the PE4302. The six stages of the RF step attenuator are along the top, together with the DPDT analogue switches which allow each stage to be switched in or out of the signal path between the RF$_{IN}$ pin at upper left and the RF$_{OUT}$ pin at upper right.

The switches for each stage are driven by the control logic interface shown in the lower part of the diagram. The first attenuation stage reduces the signal amplitude by 16dB, while later stages reduce it by 8dB, 4dB, 2dB, 1dB and 0.5dB respectively.

Since these figures are all powers of 2, this allows the chip to be programmed in binary fashion to provide any desired nominal level of attenuation between 0dB and –31.5dB.

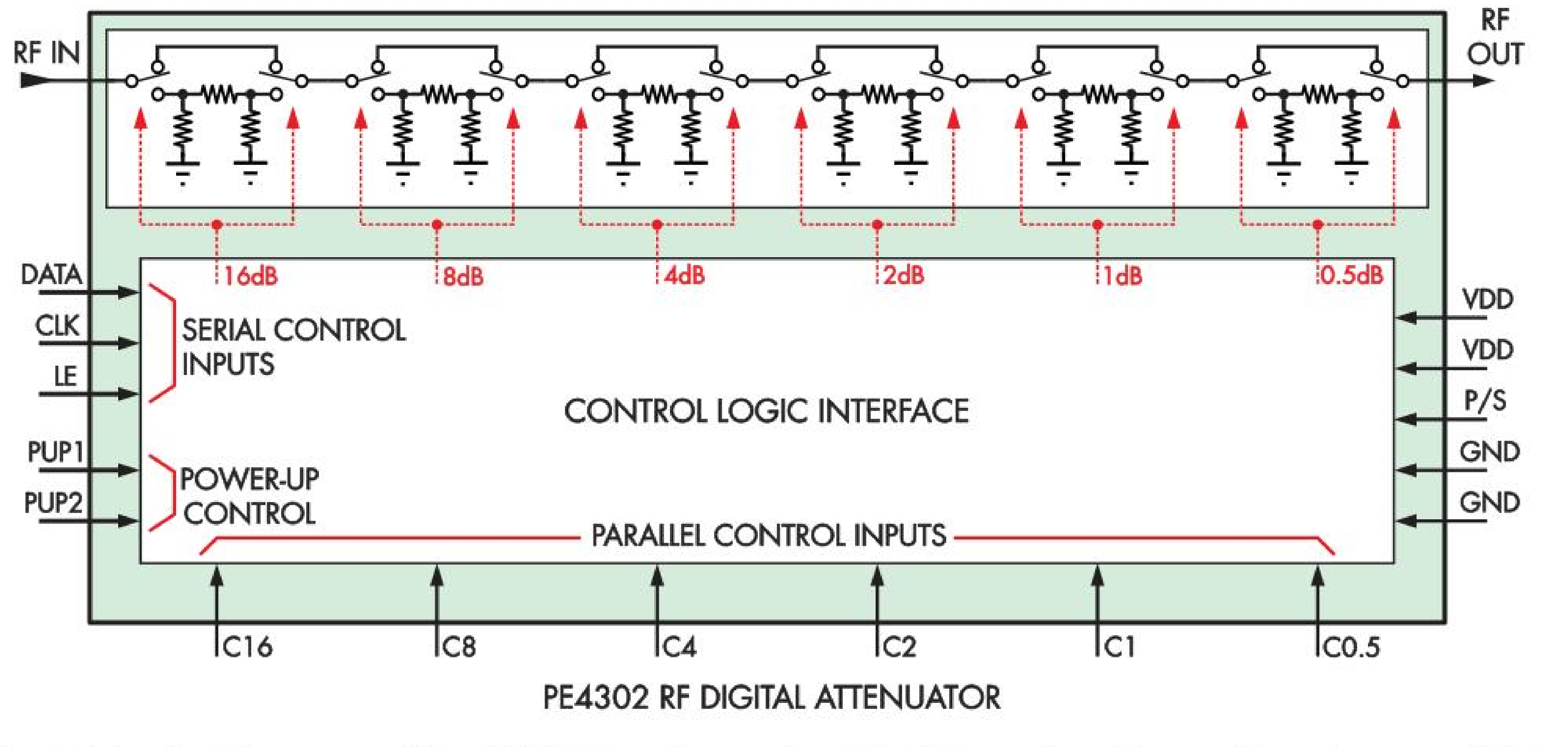


Fig.1: block diagram of the PE4302 attenuator IC. It has six attenuation stages which can be switched in or out by matching DPDT analogue switches. Serial and parallel control inputs are both provided by the IC, but the serial inputs are disabled on the PE4302 module.

In practice, the chip has an insertion loss even at the 0dB setting (with all stages switched out).

Typically this insertion loss varies between –1dB and –1.5dB at frequencies below 2.5GHz, increasing to about –3.5dB at 4.0GHz. Oddly, this only seems to apply when there is no attenuation requested. As a result, the attenuation at a setting of –0.5dB is actually considerably lower than at 0dB for all the frequencies we tested!

The control logic interface provides parallel control inputs (along the bottom) and serial control inputs (centre left). That actually gives the PE4302 three different ways of setting the attenuation level.

Setting attenuation

The first way of setting attenuation level is via direct parallel programming, where you apply logic-level signals directly to the C16-C0.5 inputs with a microcontroller or a set of DIP switches.

The second mode is latched parallel programming, where the control signals are still applied to the C16-C0.5 input pins but the LE (load enable) pin must be pulsed low when they are changed because the control signals are stored in a latch register when the LE pin is returned to logic high.

The third mode is serial programming, where the six programming bits are fed into the chip via the CMOS-compatible DATA and CLK serial interface pins, with the LE pin then pulsed high and low to store the bits in the latch. If the P/S pin is pulled high, the chip powers up in serial programming mode; otherwise, it powers up in parallel mode.

When parallel mode is enabled, the PUP1 and PUP2 pins at lower left in Fig.1, together with the LE pin at centre left, are used to determine the chip's control settings when it is powered up.

By varying the logic levels on these pins you can ensure that the chip powers up at 0dB attenuation (insertion loss only), 8dB, 16dB or 31dB, or whatever attenuation is programmed by pins C16-C0.5.

Fig.2 shows the complete circuit of the step attenuator module, and it is set in direct parallel programming mode. This has the LE pin (5) tied to the positive supply rail, while the P/S, PUP1 and PUP2 pins (13, 7 and 8) are all tied to ground, along with the DATA (3) and CLK (4) serial interface pins.

The RF input connector is coupled to pin 2 of IC1 via a 100nF capacitor, while the output is taken from pin 14 to the RF output connector via another 100nF capacitor.

The parallel programming pins C8 (15), C4 (16), C2 (17), C1 (19) and C0.5 (20) are each connected to pins V5-V1 on the 7-way SIL programming connector CON1, as well as being pulled to ground (logic low level) via 10kΩ resistors. The C16 (1) programming pin is connected in a similar fashion to pin V6 of CON1, although not directly but via a series 10kΩ resistor.

This is in line with Peregrine's recommendation, to prevent resonance effects within the chip due to the proximity of this pin to the RF_{IN} pin (2).

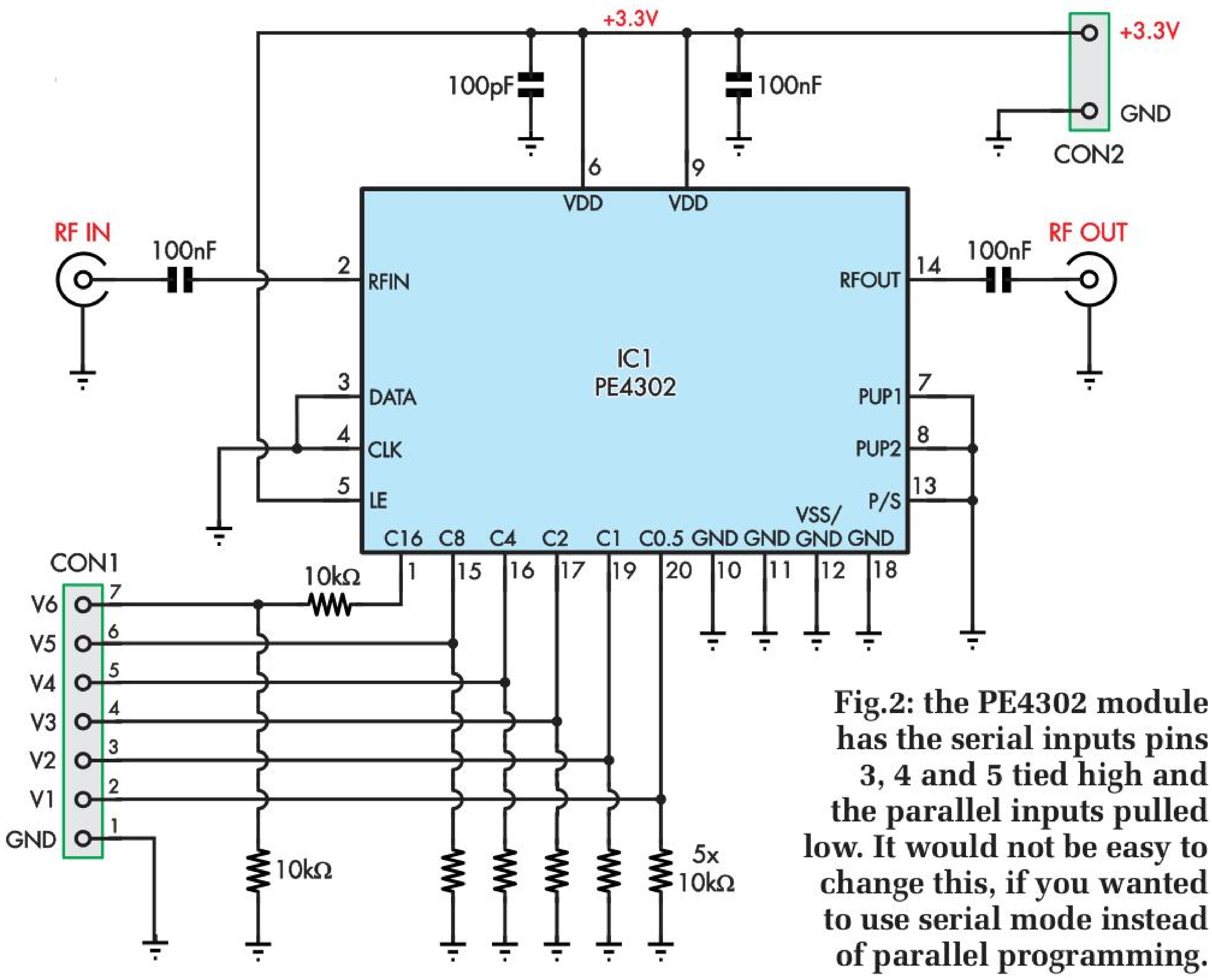


Fig.2: the PE4302 module has the serial inputs pins 3, 4 and 5 tied high and the parallel inputs pulled low. It would not be easy to change this, if you wanted to use serial mode instead of parallel programming.

Putting it to use

The simplest way to control this module is to use a 6-pole DIP switch, as shown in Fig.3. One side of each switch is connected to the +3.3V supply line so that when each switch is closed, the respective pin of CON1 will be pulled high.

The truth table to the right of Fig.3 shows some examples of the switch combinations and the resulting nominal attenuation settings.

You could use a similar approach to control the module directly from a micro, like an Arduino or a Micromite. In which case, you'd power the module from the +3.3V and GND pins of the micro, and connect programming pins V1-V6 to six spare digital I/O pins on the micro.

Then it would be a matter of writing a program to control the attenuator module via these six pins.

The difficulty with this approach is that you may not have six spare I/O pins available. Unfortunately, as noted, the module is hard-wired for parallel programming, with the serial interface effectively disabled.

Luckily, there is a way to work around this so you can control the module from your micro via a standard I^2C serial interface. That's by making use of one of the very low cost piggyback serial interface modules, based on either the PCF8574T chip or its sibling, the PCF8574AT.

These modules are intended to adapt a parallel-interface LCD module for serial interfacing and they often come mated with an LCD.

But they are also available separately for £1 or less each, and this makes them very attractive for solving other I^2C/parallel interfacing jobs like this one.

Fig.4 shows how to use one of these PCF8574T/AT modules to connect up the PE4302 digital step attenuator module to your micro for serial control via the I^2C bus.

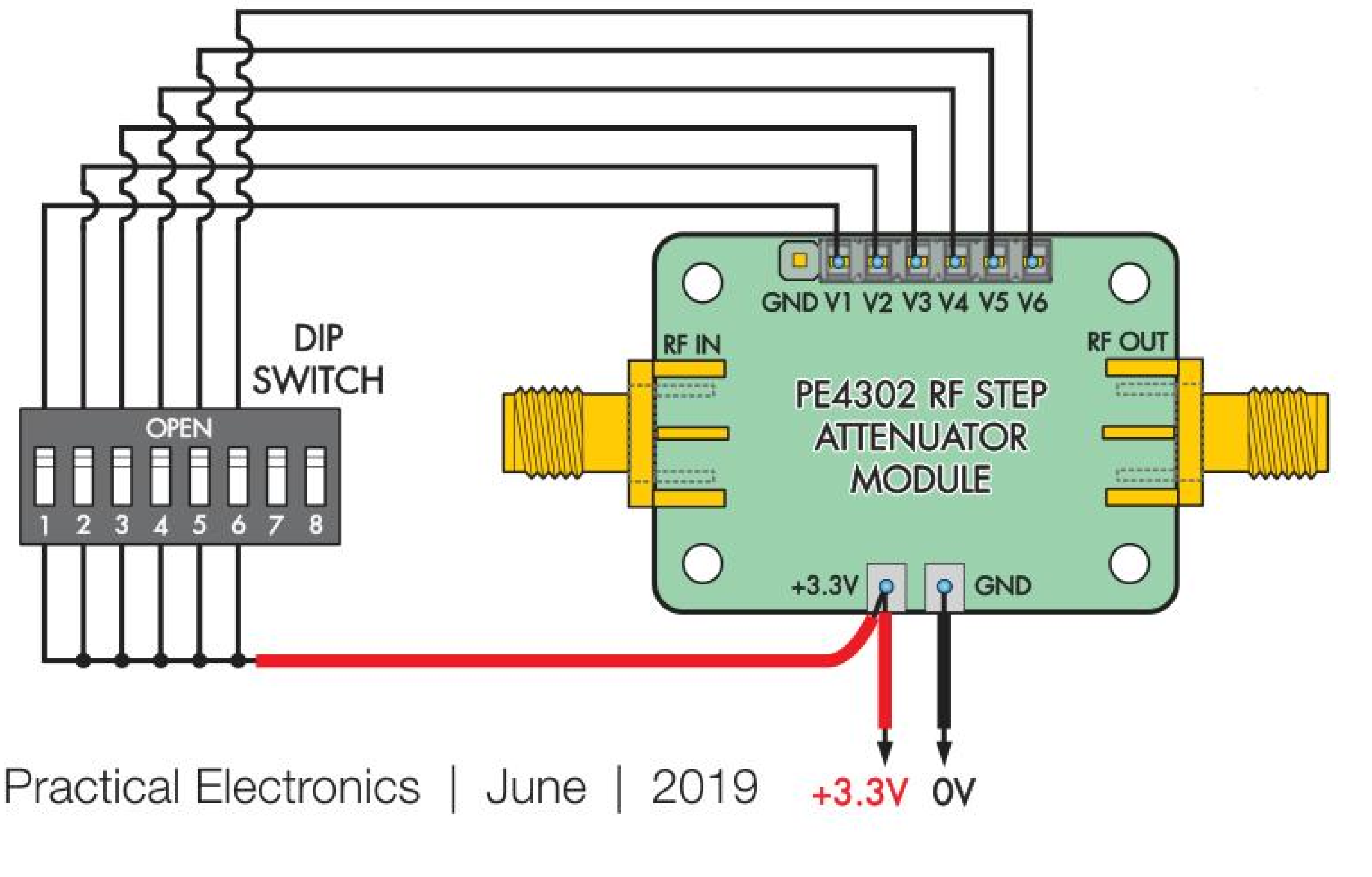
The upper part shows the circuitry inside the piggyback module, while the PE4302 module is shown at lower right, with the interconnections all made via the 16-pin header which usually connects to the LCD module.

The six programming lines pass through IC2, a 74HC367 hex non-inverting buffer. This is needed because the outputs from IC1 can only provide very low current in their high logic state, but the V1-V6 inputs of the step attenuator module are all fitted with 10kΩ pulldown resistors. This means that they tend to draw more current than the outputs of IC1 can provide.

Alternatively, you could leave out the 74HC367 and simply remove the six pull-down resistors from the underside of the module.

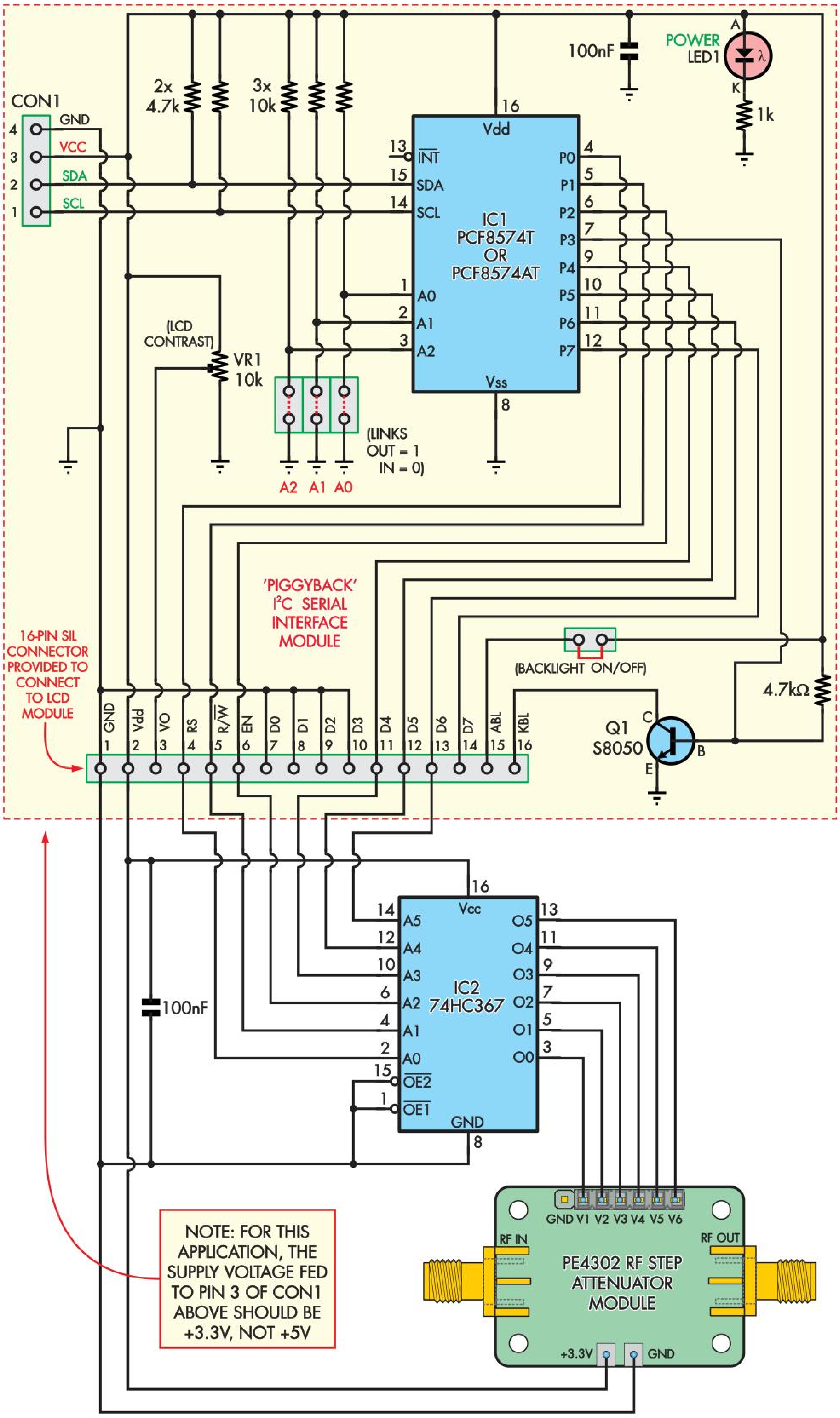
Notice that we've also shown a table at lower left in Fig.4 with the various I^2C addresses applying to the

Fig.3: manual programming can be done with a 6-pole DIP switch attached to the PE4302 module. The table below shows some of the switch combinations and the resultant attenuation settings.

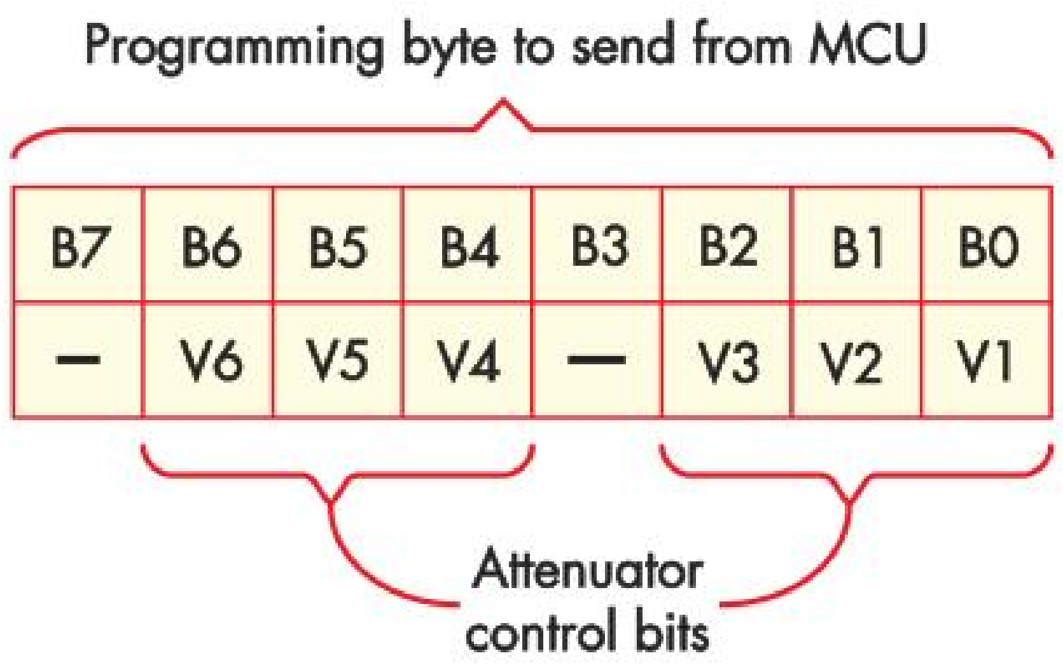


PE4302 Programming Truth Table

V1 (C0.5)	V2 (C1)	V3 (C2)	V4 (C4)	V5 (C8)	V6 (C16)	ATTENUATION
0	0	0	0	0	0	(Insertion loss only)
1	0	0	0	0	0	0.5dB
0	1	0	0	0	0	1.0dB
0	0	1	0	0	0	2.0dB
0	0	0	1	0	0	4.0dB
0	0	0	0	1	0	8.0dB
0	0	0	0	0	1	16dB
0	1	0	1	0	1	21dB
1	1	1	1	1	1	31.5dB


CON1
GND
VCC
SDA
SCL
4
3
2
1
2x 4.7k
3x 10k
100nF
POWER LED1
A
K
λ
1k
16
Vdd
13 INT
15 SDA
14 SCL
IC1 PCF8574T OR PCF8574AT
P0 4
P1 5
P2 6
P3 7
P4 9
P5 10
P6 11
P7 12
1 A0
2 A1
3 A2
Vss
8
(LCD CONTRAST)
VR1 10k
(LINKS OUT = 1 IN = 0)
A2 A1 A0
'PIGGYBACK' I²C SERIAL INTERFACE MODULE
16-PIN SIL CONNECTOR PROVIDED TO CONNECT TO LCD MODULE
(BACKLIGHT ON/OFF)
4.7kΩ
Q1 S8050
C
B
E
GND Vdd VO RS R/W EN D0 D1 D2 D3 D4 D5 D6 D7 ABL KBL
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
16
Vcc
14 A5
12 A4
10 A3
6 A2
4 A1
2 A0
15 OE2
1 OE1
IC2 74HC367
O5 13
O4 11
O3 9
O2 7
O1 5
O0 3
GND
8
100nF
GND V1 V2 V3 V4 V5 V6
RF IN
RF OUT
PE4302 RF STEP ATTENUATOR MODULE
+3.3V
GND
NOTE: FOR THIS APPLICATION, THE SUPPLY VOLTAGE FED TO PIN 3 OF CON1 ABOVE SHOULD BE +3.3V, NOT +5V

I²C ADDRESSING FOR THE PCF8574T & THE PCF8574AT				
A2	A1	A0	PCF8574T ADDRESS	PCF8574AT ADDRESS
1	1	1	27	3F
1	1	0	26	3E
1	0	1	25	3D
1	0	0	24	3C
0	1	1	23	3B
0	1	0	22	3A
0	0	1	21	39
0	0	0	20	38


Programming byte to send from MCU

B7	B6	B5	B4	B3	B2	B1	B0
–	V6	V5	V4	–	V3	V2	V1

Attenuator control bits

Fig.4: wiring diagram for the PE4302 module connected to a serial I²C 'piggyback' module (IC1) via a hex non-inverting buffer (IC2). At bottom right is the format of the byte to be sent from the micro to the PE4302 to activate each attenuator. Ergo, 01110110 (most-significant-bit first) activates C1, 2, 4, 8 and 16 (but not C0.5).

piggyback module, depending on (a) whether it's using a PCF8574T chip or a PCF8574AT chip, and (b) whether any of the onboard links A0, A1 or A2 are shorted.

If you're in any doubt regarding which of the two chips is fitted to your module, this can usually be clarified quite easily by examining the top of the chip with a magnifying glass.

Note that because the PE4302 chip must be connected to a +3.3V supply, this also means that pin 3 of CON1 on the piggyback module should be connected to +3.3V rather than the usually expected +5V. This won't be a problem for the piggyback module because both versions of the PCF8574T/AT are designed to work from any supply voltage between 2.5V and 6V.

As shown in Fig.4, you connect pins V1-V3 of the attenuator module to pins 4-6 on the piggyback module (via IC2), while pins V4-V6 are connected to pins 11-13.

As a result, to program the attenuator correctly you simply need to send it the six control bits embedded in a single byte, as shown at bottom right.

Note that bits B3 and B7 are not used and can be left at either zero or one.

Performance testing

I measured the performance of the digital step attenuator with my VHF/UHF signal generator and power meter. Because of the larger number of possible attenuation factors, this inevitably took rather long, even though I elected to do measurements for only 12 of the 64 combinations of programming bits.

I took measurements at eight different frequencies: 100MHz, 1.0GHz, 1.5GHz, 2.0GHz, 2.5GHz, 3.0GHz, 3.5GHz and 4.0GHz.

Note that the measurements were taken at nominal attenuation factors of 0dB (ie, insertion loss only), –0.5dB, –1.0dB, –2.0dB, –4.0dB, –8.0dB, –16.0dB, –20.0dB, –24.0dB, –28.0dB, –30.0dB and –31.5dB. These were chosen to give a good idea of the module's overall performance.

The results are quite close to the nominal values shown in the left-hand column of Table 1. For example, the measured value for a nominal attenuation factor of –16.0dB at 1.5GHz turns out to be –16.06dB; pretty darn close.

Similarly, the measurement at 3.5GHz for a nominal attenuation of –8dB proved to be –7.95dB; again, not far off.

Overall, the performance is quite good, at least for frequencies up to about 1.5GHz, but at higher frequencies, the relative accuracy does seem to deteriorate somewhat.

I suspect that there are two reasons for this. The first is that the open construction of the module probably allows some of the RF input signal to 'jump over' the PE4302 chip package, especially at frequencies of 2.0GHz and above (ie, due to the effects of stray capacitance).

The other likely reason is that the input and output impedances of the PE4302 almost certainly vary from 50Ω

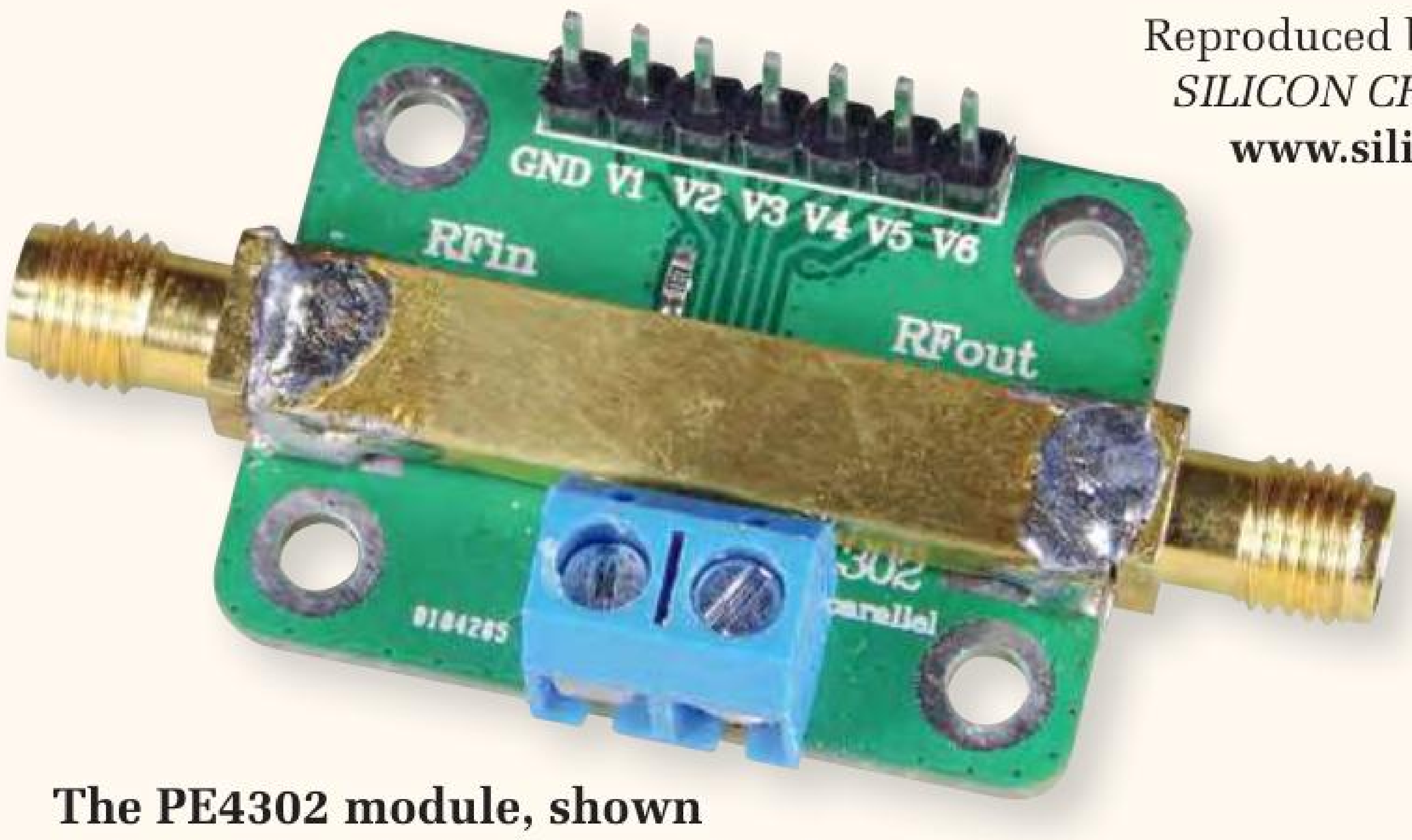
TABLE 1: MEASURED PERFORMANCE OF PE4302–BASED DIGITAL STEP ATTENUATOR

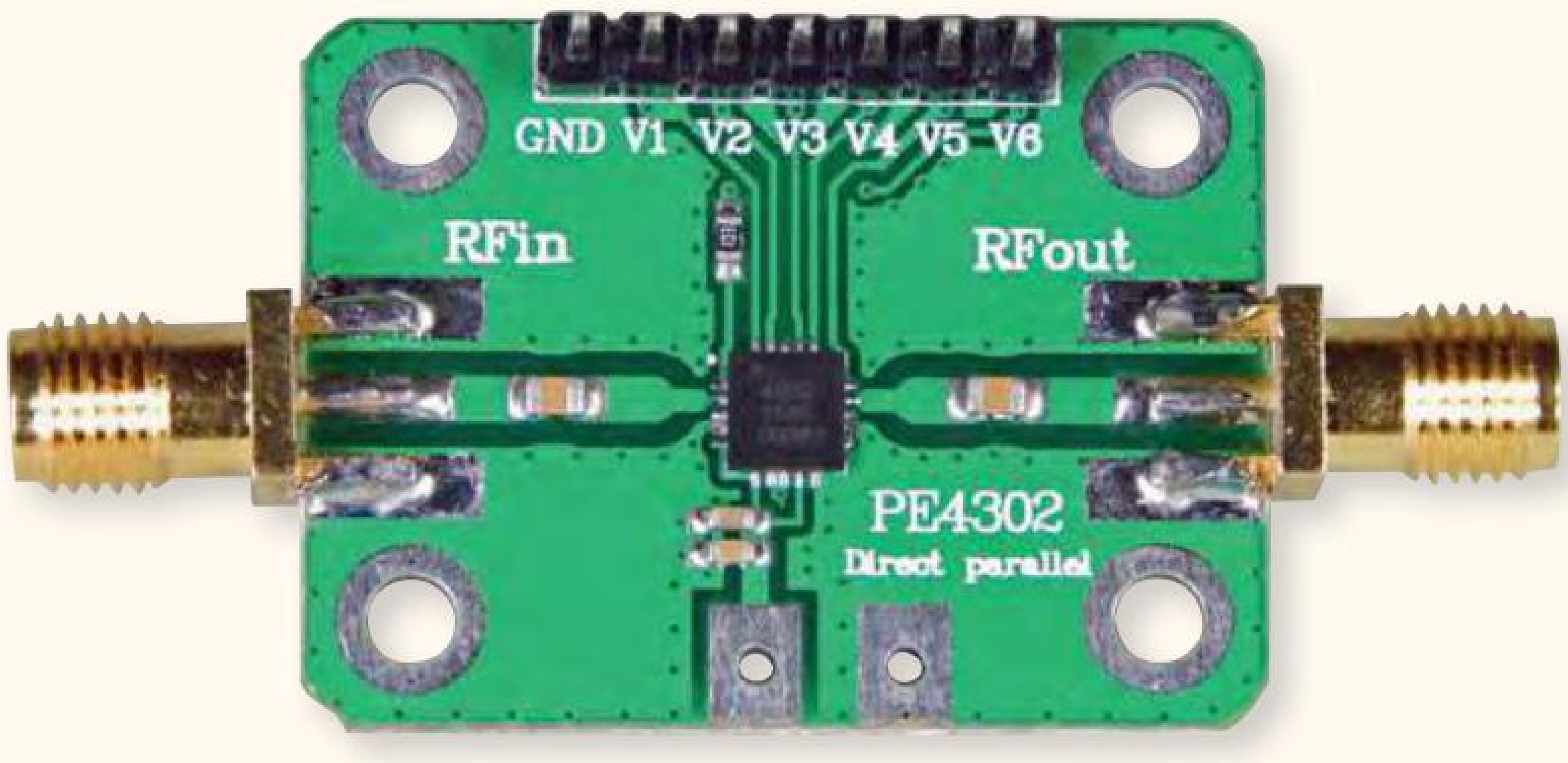
NOMINAL ATTENUATION	MEASURED RELATIVE ATTENUATION IN DECIBELS (excluding cable losses)							
	@100MHz	@1.0GHz	@1.5GHz	@2.0GHz	@2.5GHz	@3.0GHz	@3.5GHz	@4.0GHz
0dB	–1.22	–1.51	–1.80	–1.80	–1.37	–2.76	–5.30	–5.22
–0.5dB	–0.53	–0.50	–0.38	–0.60	–0.75	–0.78	–0.76	–0.71
–1.0dB	–1.01	–0.98	–0.90	–1.18	–1.36	–1.30	–1.11	–1.08
–2.0dB	–2.08	–2.09	–2.00	–2.32	–2.76	–1.37	–2.19	–1.97
–4.0dB	–4.12	–4.12	–4.05	–4.42	–5.02	–4.76	–3.98	–3.72
–8.0dB	–8.08	–8.04	–7.99	–8.29	–9.15	–8.81	–7.95	–6.92
–16.0dB	–16.08	–16.08	–16.06	–17.11	–17.54	–17.31	–16.19	–12.20
–20.0dB	–20.09	–20.06	–19.87	–19.57	–21.26	–21.04	–18.88	–14.16
–24.0dB	–24.01	–23.74	–23.14	–22.60	–24.20	–23.52	–20.05	–24.53
–28.0dB	–28.03	–27.37	–26.20	–25.31	–26.77	–25.49	–20.68	–15.47
–30.0dB	–30.02	–29.12	–27.24	–26.44	–27.81	–26.38	–20.99	–16.26
–31.5dB	–31.51	–30.22	–28.02	–27.57	–28.40	–26.75	–21.28	–16.67

TABLE 2: MEASURED PERFORMANCE OF PE4302–BASED DIGITAL STEP ATTENUATOR (with shield)

NOMINAL ATTENUATION	MEASURED RELATIVE ATTENUATION IN DECIBELS (excluding cable losses)							
	@100MHz	@1.0GHz	@1.5GHz	@2.0GHz	@2.5GHz	@3.0GHz	@3.5GHz	@4.0GHz
0dB	–1.25	–1.57	–1.62	–1.73	–1.30	–2.31	–4.35	–6.45
–4.0dB	–4.11	–4.13	–4.23	–4.53	–4.95	–4.88	–4.16	–4.22
–8.0dB	–8.04	–8.01	–8.18	–8.54	–9.15	–9.05	–8.21	–8.72
–16.0dB	–16.06	–16.72	–16.38	–16.64	–17.71	–17.95	–18.03	–18.98

Reproduced by arrangement with
SILICON CHIP magazine 2019.
www.siliconchip.com.au





The PE4302 module, shown enlarged for clarity. Without and with the earthed metal shield.

at these higher frequencies, causing standing waves in the cables.

In fact, Peregrine Semiconductor's data sheet shows both input and output return loss plots varying quite widely over the full frequency range. Both rise significantly at higher frequencies.

There's probably not much that can be done about the cable matching/standing waves problem since it's inherent in the chip itself. Not content with that, I decided to try improving the overall attenuation accuracy at the higher frequencies by fitting an earthed metal shield over the PE4302 chip and its input and output coupling capacitors.

The shield is 33 × 7 × 2.5mm and was soldered at each end to the earthed outer frame of the SMA connectors.

Table 2 shows the modest improvements after the shield was fitted, so it is probably a worthwhile exercise for very little effort.

PoLabs

For more information or
software download please visit

www.poscope.com/epe

PoKeys
Connect, Control

- USB
- Ethernet
- Web server
- Modbus
- CNC (Mach3/4)
- IO
- PWM
- Encoders
- LCD
- Analog inputs
- Compact PLC

Stepper motor drivers
Drive

- up to 256 microsteps
- 50 V / 6 A
- USB configuration
- Isolated
- up to 32 microsteps
- 30 V / 2.5 A

PoScope Mega1+
PoScope Mega50
Measure

- up to 50MS/s
- resolution up to 12bit
- Lowest power consumption
- Smallest and lightest
- 7 in 1: Oscilloscope, FFT, X/Y, Recorder, Logic Analyzer, Protocol decoder, Signal generator

MIDI Ultimate Synthesiser

MIDI Ultimate
Soundtronics
MFOS

Introduction to the Musical Instrument Digital Interface (MIDI)

Part 5 | by Paul Cooper | soundtronics.co.uk

In Part 5 of the MIDI Ultimate Synthesiser build, we are going to provide a brief overview of the Musical Instrument Digital Interface (MIDI). It has been some time since MIDI was covered as a topic in *PE*, so this month's article on the *MIDI Ultimate* differs slightly. We will start with a general interest introduction to serial communications using the MIDI interface and follow that with how it is implemented within the *MIDI Ultimate* synth.

Many of us 'old hands' have been working with MIDI for more years than we may care to remember; but despite approaching its 40th anniversary, MIDI remains very relevant today. This article is more of a general introduction to what makes up MIDI – **www.midi.org** is the site to visit should you need further detailed information, especially with regards to the particular messages used in the protocol.

Brief history of MIDI

MIDI, short for 'Musical Instrument Digital Interface', emerged in 1981/82 following collaboration between the large synth manufacturers of the time, including Roland, Yamaha, Korg Sequential Circuits and Kawai. Commercial products implementing MIDI then became available from 1982 onwards. MIDI was developed to allow for a common interface between different electronic musical instruments, thereby standardising manufacturers' proprietary standards. In 2019, a quick look around various synth manufacturers' websites reveals that the MIDI interface is still implemented in their latest product ranges. Is MIDI the perfect solution that cannot be bettered? Actually no, it is very dated in many aspects, but because it has become so established and universally adopted, attempts to replace it have so far had limited success.

Communication standards typically (but not always) define the electrical, mechanical and software protocol of the interface, and MIDI is no different. The electrical specification that we shall cover later is largely unchanged since 1982. The mechanical interface, by which we mean the physical connectors, has traditionally been the 5-pin DIN connector (aka 'over-DIN') but in 2018, a new standard was published that covered the additional use of 2.5mm TRS jack connectors. MIDI information can also be sent over USB, FireWire, Ethernet, Bluetooth and more, but USB is the most popular means for interfacing your electronic musical instrument to a computer.

The MIDI protocol has been enhanced over the years, but it is still called MIDI 1.0, as published in 1996 by the non-profit MIDI Manufacturers Association (MMA). The MMA has a

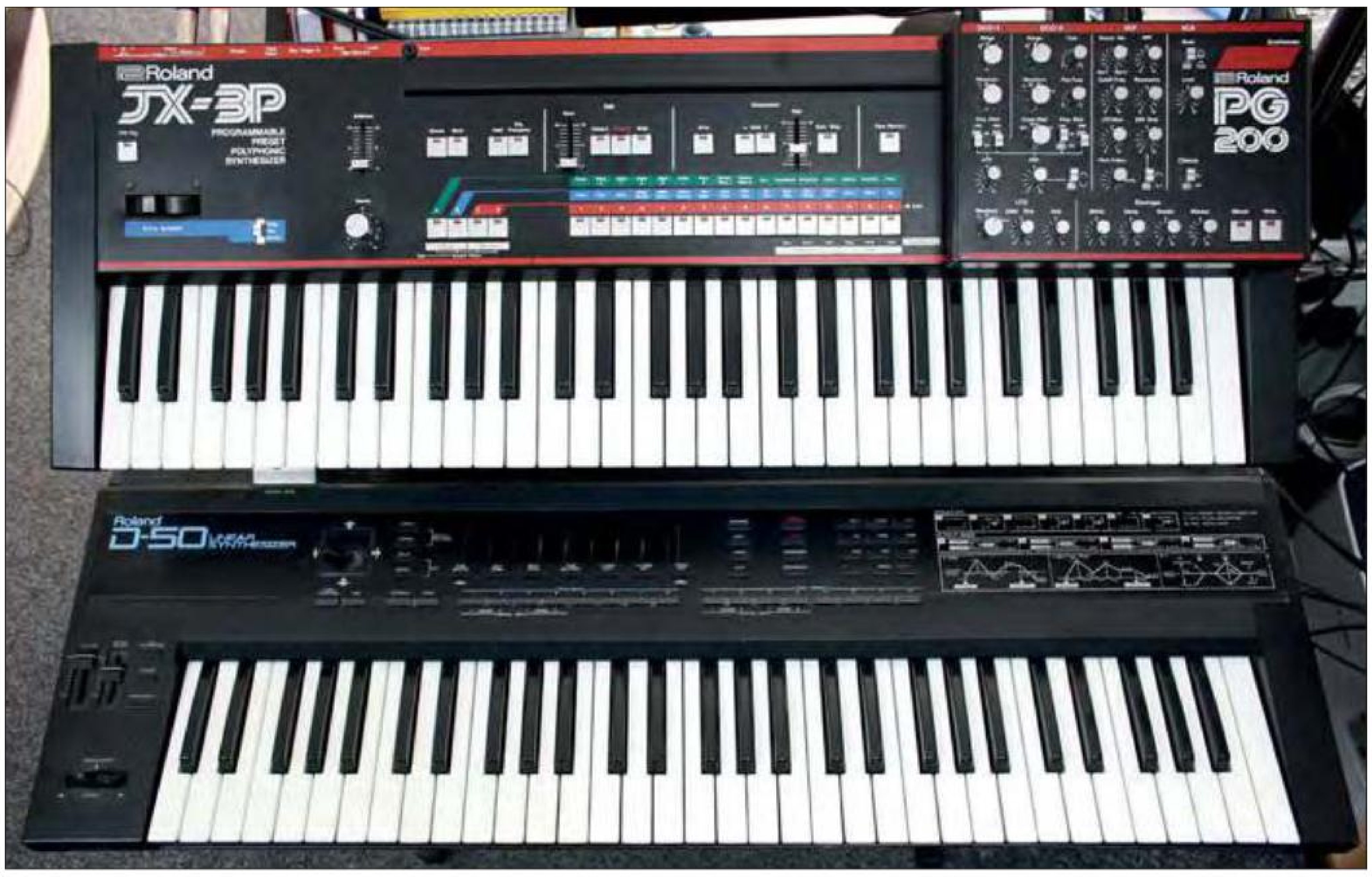


Fig.71. The Authors mono-timbral synths – each synth uses a single MIDI channel.

website (**www.midi.org**) where the fully detailed specifications for the MIDI interface can be found, plus lots more MIDI-related topics. It's the home for all things MIDI – from legacy equipment to next-generation protocols.

The future for MIDI as we know it today remains uncertain; MIDI HD is being discussed but that is likely to only be successful if backward compatibility to MIDI 1.0 is maintained. Open Sound Control (OSC) is a modern alternative to MIDI – it has yet to make much impact on the hardware market, but it has been adopted by many software-based applications.

What is MIDI?

Let's start with what MIDI isn't. MIDI cannot create or carry audio or video signals of any type. MIDI is a specification for the connecting and controlling of electronic musical instruments – a music description language in digital form. It can also be used for other applications which are beyond the scope of this article. It is a unidirectional asynchronous data stream that sends data one bit at a time at 31.25kb/s which form information and instructions for the receiving device or instrument. Putting this last sentence into context, MIDI can only transmit in one direction at 31.25kbits/s, or 3125 bytes/s – a MIDI byte requires ten bits for transmission. Compare this with USB 3.0's 625MB/s and MIDI is 200,000 times slower! Remember, MIDI dates back to 1982 at a time when the BBC Micro was launched with 16KB of memory. Fortunately, the relatively slow data transmission rate of MIDI is not such a problem when the amount of data being transmitted is small and concise. For example, a 24-minute MIDI file of Mike Oldfield's *Tubular Bells* has a file size of around 320kB covering 37 instrument tracks, which equates to just 13kB of data per minute. Transmitting a 'Note On' message when a key on the keyboard is pressed happens in around 1ms over-DIN. However, simultaneously press eight keys in one go and it will take nearer 8ms to send.

MIDI channels and messages

Before describing the MIDI message types, it's important to note that MIDI is not limited to communications between two devices. One MIDI link or connection can carry 16 channels or streams. This means up to 16 separate mono-timbral instruments (instruments that can only produce one sound or patch at a time; ie, the examples in Fig.71.). Or, it means up to 16 sounds or patches on a multi-timbral instrument at once (ie, the device in Fig.72.) from a single central device such as a MIDI controller or computer. These channels are referenced as 1-16, but in the protocol they are numbered 0-15.

There are two main types of MIDI message, those that relate to a specific channel – *channel* messages, and those that are independent of any channel – *system* messages. These primary message types are sub-divided as per Table 5.

Before we delve deeper into MIDI messages, let's take a step back and cover the electrical and mechanical standards.

MIDI connectors

Most electronic musical instruments and equipment that have a MIDI interface will most likely be provided with 5-pin DIN connectors, but more modern instruments will also have a USB connector (Fig.73). There are usually between one and three DIN connectors, although some sequencers may have more. If the equipment is a keyboard controller with no capability of producing sound, it will have a MIDI Out socket. A synth will have at least a MIDI In socket, and if it has an integral keyboard them expect a MIDI Out socket so it can control other synths and finally, a MIDI Thru socket that re-transmits the messages appearing at the MIDI In socket. The MIDI Thru socket can feed the next instrument's MIDI In socket, so you can daisy chain your instruments on the same MIDI link, but set to different MIDI channels.

Daisy chaining MIDI devices can introduce what is termed latency, although not all MIDI users experience this, and it can be manufacturer dependent. Latency is where there is a delay between the message appearing at the MIDI In socket and repeated at the MIDI Thru

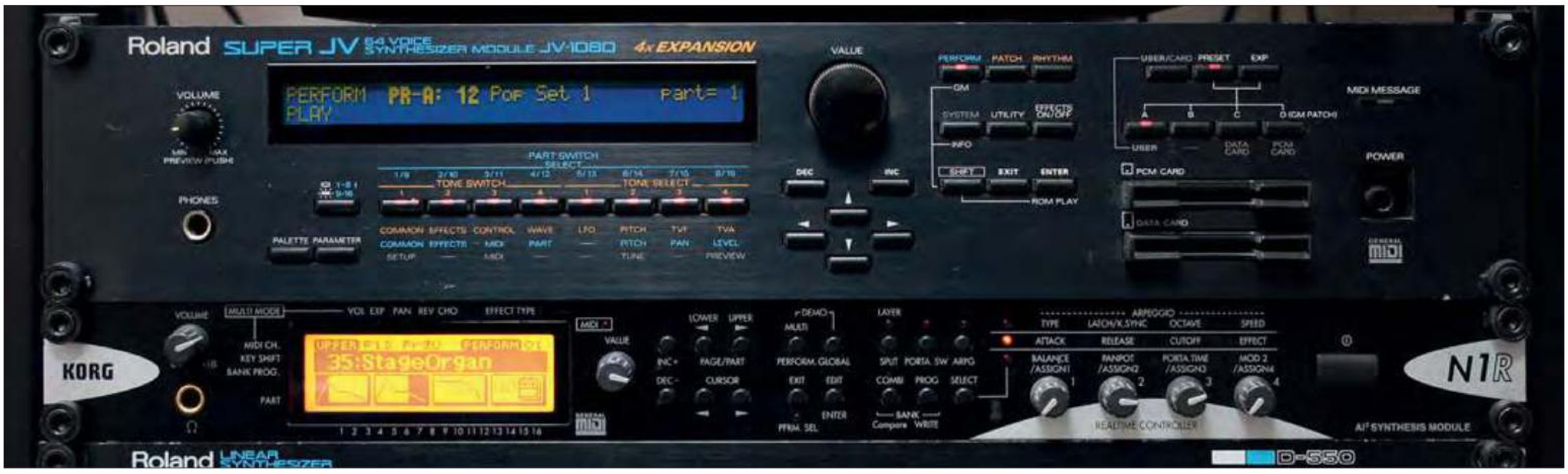

Roland SUPER JV 64 VOICE SYNTHESIZER MODULE JV-1080 4x EXPANSION
PERFORM PR-A: 12 Pop Set 1 Part= 1
PLAY
KORG
35:StageOrgan
N1R
Roland LINEAR SYNTHESIZER
D-550

Fig.72. The authors two multi-timbral synths that can each play up to 16 different instruments or patches simultaneously, using up to 16 MIDI channels per synth.

Table 5: MIDI message types and examples of use

Message type	**Description**	**Examples**
Channel Voice	**Messages that carry musical performance data and typically account for most traffic data**	**Note on/off, key velocity, pitch bend**
Channel Mode	**Messages that affect the way the receiving device responds to channel voice messages**	**All sound off, poly mode on**
System Real-Time	**Used to synchronise all of the MIDI clock-based systems and has the highest priority and can occur at any time in the message stream**	**Synchronise sequencers and drum machines**
System Common	**More advanced instructions for timing, song position pointer and tune requests**	**Song position pointer, song select on sequencers and drum machines**
System Exclusive (SysEx)	**Equipment manufacturers can define their own messages to control features within their equipment as well as some reserved for special protocols**	**Send patch data or even firmware updates**

socket. While the delay can be just a few milliseconds, this can compound to a noticeable delay if you have a long daisy chain of instruments. One solution to this is to use MIDI Thru boxes which have a MIDI In socket with multiple MIDI Thru sockets so now you can parallel up your instruments and minimise any latency. Fig.74 is the MIDI arrangement I use and includes MIDI switches and MIDI Thru boxes

The wiring to the MIDI In and Out connectors can cause confusion because the wiring differs between them, MIDI In *receives* data, MIDI Out *transmits* data. The *MIDI Ultimate* uses the DIN sockets as shown in Fig.75, but many other types are available for both panel and PCB mounting.

MIDI signal conditioning

The MIDI specification describes the electrical specification for the signal conditioning of the MIDI sockets. While the DIN sockets are 5-pin, pins 1 and 3 are not used at any time. Linking together multiple electronic musical instruments with DIN cables can introduce problems such as ground loops, resulting in a degradation of signal quality and range. When correctly implemented, in accordance with MIDI 1.0, this is avoided through the use of opto-isolation and signal buffering. Fig.76 is a typical schematic for each of the MIDI sockets and has been used successfully by Soundtronics.

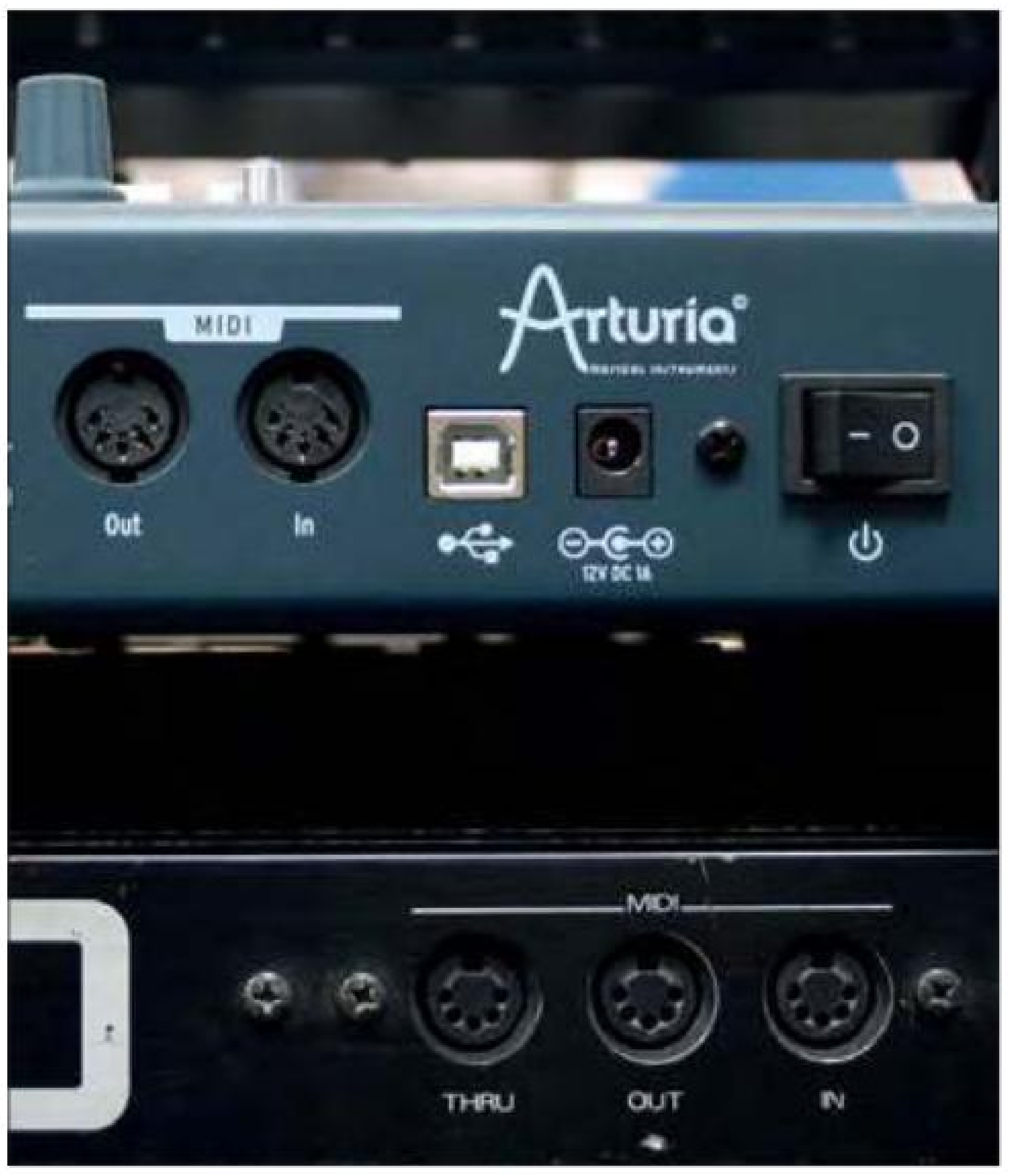


Fig.73. MIDI sockets on commercial synths, the D50 (below) from 1987 and the Minibrute (above) from 2012, along with an additional USB port.

Sparkfun.com in the US has a variety of MIDI Arduino shield PCBs that are very useful in experimenting with MIDI using an Arduino UNO. MIDI leads are available from many suppliers including Soundtronics and are generally not worth going to the trouble to make up yourself. MIDI has its limitations in transmission distances and any one cable should not be longer than 15m. The latest update to the electrical specification for MIDI 1.0 now describes the optional use of ferrite beads on the signal lines from pins 4 and 5 to improve EMI/EMC performance, although I have not yet found them necessary.

MIDI protocol

Returning to the protocol and the five message types described in Table 5, each channel message is transmitted in multiple bytes consisting of one status byte followed by one or two data bytes. The System Exclusive messages can have any number of data bytes following the status byte, and are terminated by the End of Exclusive (EOX) byte or any other status byte, excluding real-time messages. Real-time messages consist of a single status byte only (Fig.77).

A status byte has bit D7 set to ‘1’ to indicate it is a status byte and not a data byte.

Channel Voice messages

The status byte for Channel Voice messages are split into two nibbles (2 × 4-bits). The lower four bits D0 to D3 represent channel number 1-16 expressed in binary 0000 to 1111 (0-15). Bits D4 to D6 indicate the function. Consider these two examples.

When a key is depressed on a keyboard, `1001nnnn` is transmitted, where `nnnn` is the MIDI channel number and 1001 indicates it is the status byte for a note-on event. The receiving device then expects two further bytes that contain further information about the note pressed, the note number and the note key velocity. All data bytes have bit D7 set to a 0, therefore the maximum number of notes that MIDI can recognise is 128 (0-127), a piano has 88 notes so hardly a limitation. The *MIDI Ultimate* design discusses this further, later in this article.

Another example is the Control Change, which has a value of `1011nnnn`, where once again nnnn is the MIDI channel number. The status byte is also followed by two bytes, `0ccccccc` and `0vvvvvvv` where `ccccccc` is the controller number (0-119) and `vvvvvvv` is the controller value (0-127). Note controller numbers 120-127 are reserved for Channel Mode messages (see below). This Control Change message is sent when a controller value changes. Controllers are devices such as modulation wheels, breath controllers or foot controllers.

Channel Mode messages

The status byte for Channel Mode messages has a value of `1011nnnn` (like the Control Change message above) where, as before, nnnn is the MIDI channel number. The status byte is again followed by two bytes, `0ccccccc` and `0vvvvvvv` where `ccccccc` is the controller number (120-127) and `vvvvvvv` is the controller value (0-127). Controller numbers 120-127 are reserved for specific functions; for example, 123 is all notes off. Most of these reserved controllers have a third data byte of value 0.

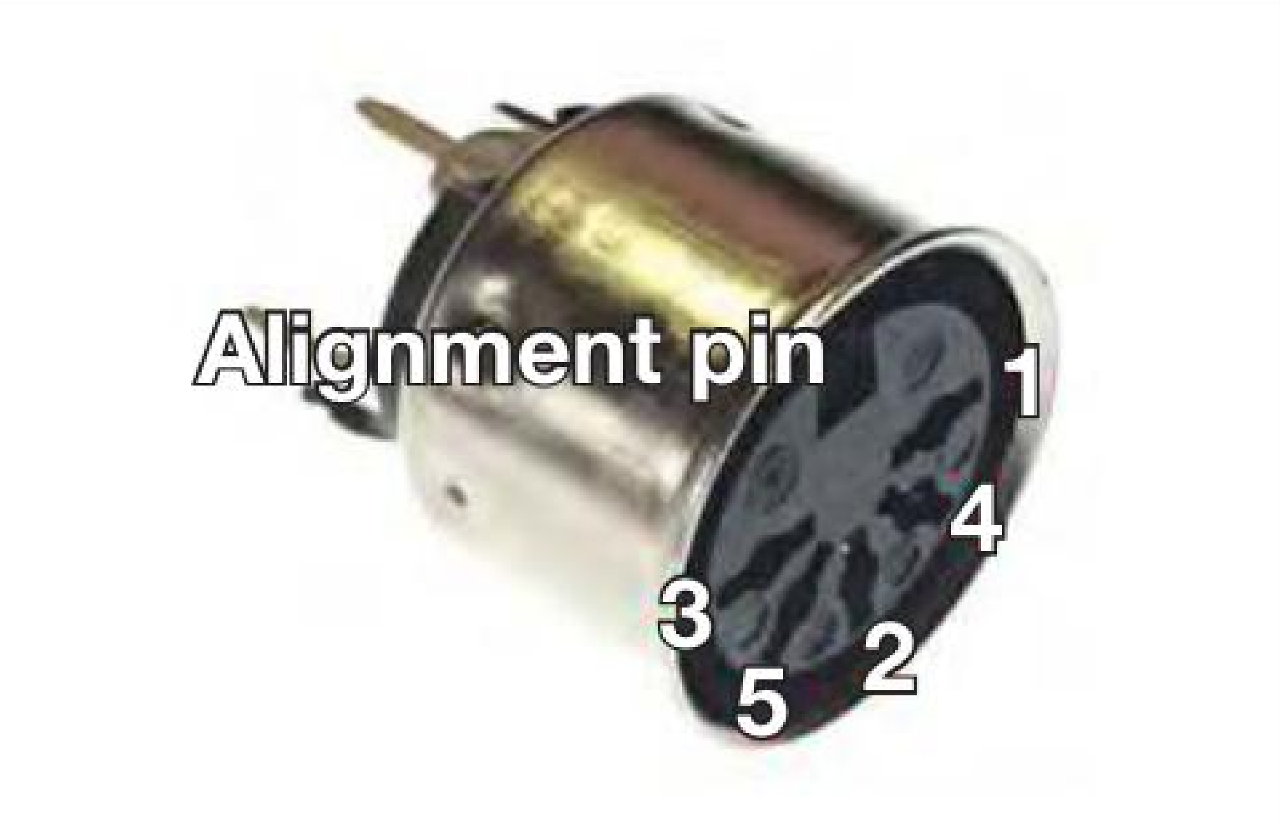


Fig.75. 5-Pin DIN socket used in the *MIDI Ultimate* for MIDI In and MIDI Thru with pin numbering.

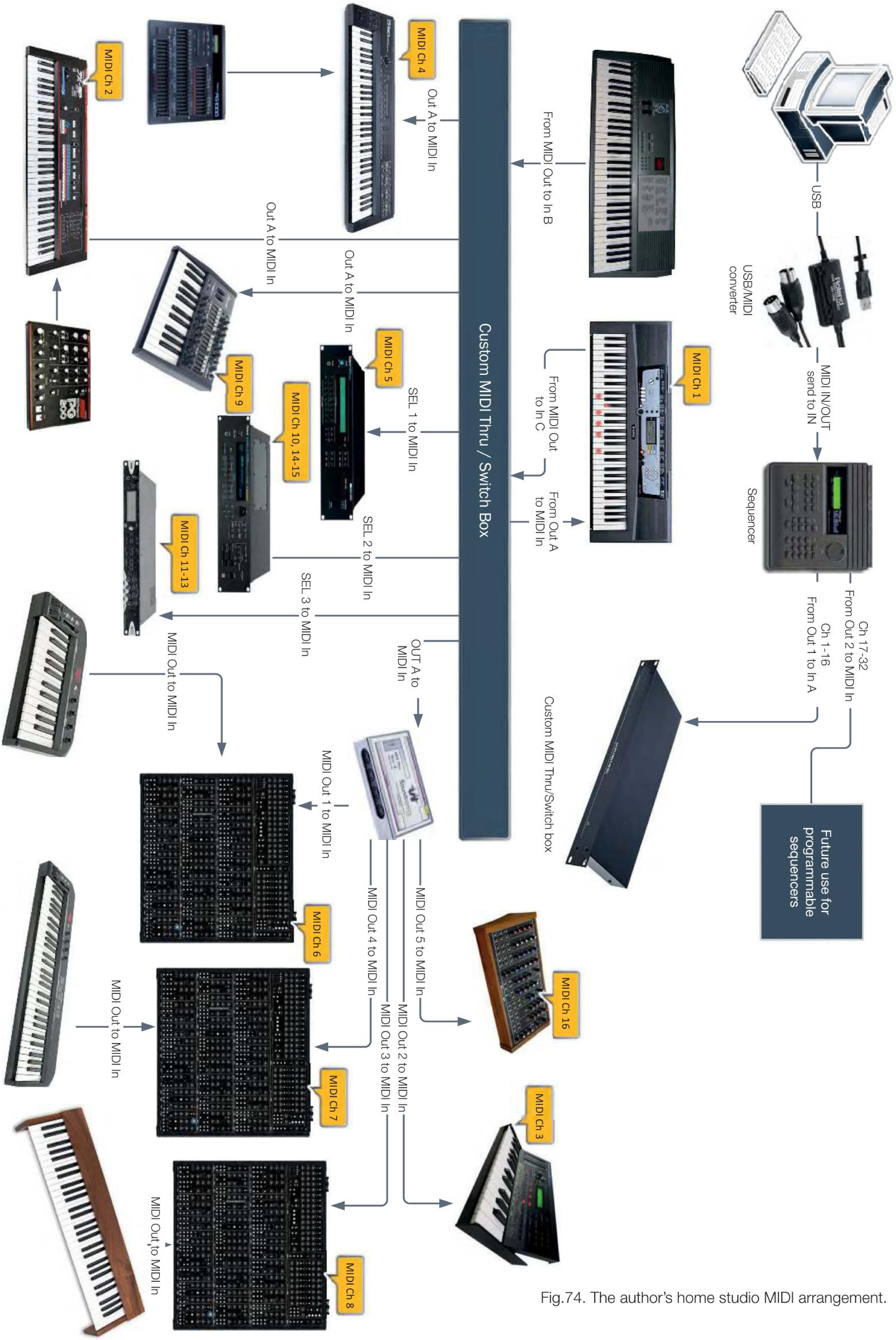

USB
USB/MIDI converter
MIDI IN/OUT send to IN
Sequencer
Ch 17-32 From Out 2 to MIDI In
Ch 1-16 From Out 1 to In A
Future use for programmable sequencers
Custom MIDI Thru/Switch box
From MIDI Out to In B
MIDI Ch 1
From MIDI Out to In C
From Out A to MIDI In
Custom MIDI Thru / Switch Box
MIDI Ch 4
Out A to MIDI In
MIDI Ch 2
Out A to MIDI In
Out A to MIDI In
MIDI Ch 9
MIDI Ch 5
SEL 1 to MIDI In
MIDI Ch 10, 14-15
SEL 2 to MIDI In
MIDI Ch 11-13
SEL 3 to MIDI In
OUT A to MIDI In
MIDI Out to MIDI In
MIDI Out 1 to MIDI In
MIDI Out 5 to MIDI In
MIDI Out 4 to MIDI In
MIDI Out 2 to MIDI In
MIDI Out 3 to MIDI In
MIDI Ch 6
MIDI Ch 16
MIDI Out to MIDI In
MIDI Ch 7
MIDI Ch 3
MIDI Out to MIDI In
MIDI Ch 8

Fig.74. The author's home studio MIDI arrangement.

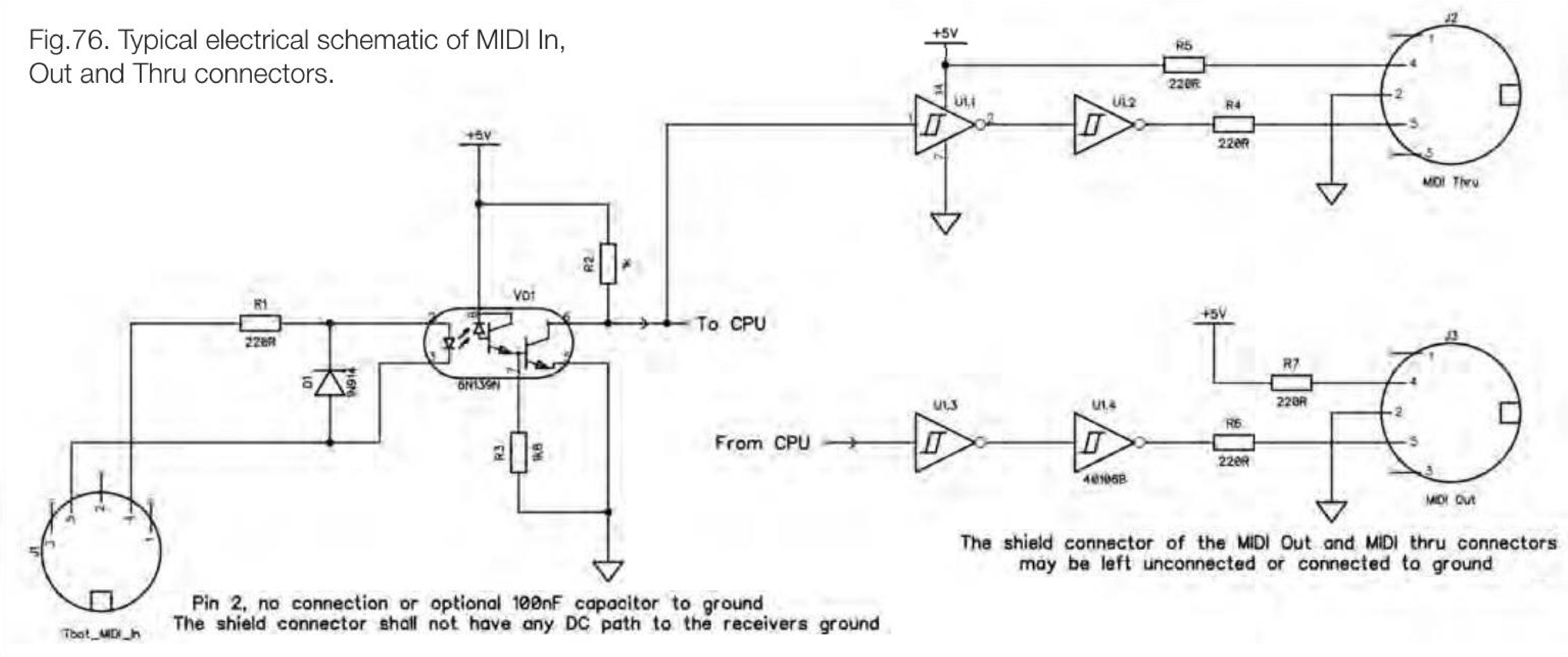


Fig.76. Typical electrical schematic of MIDI In, Out and Thru connectors.

+5V
R5
220R
J2
U1.1
U1.2
R4
220R
MIDI Thru
+5V
R2
VO1
R1
220R
To CPU
D1
1N914
6N139N
R3
+5V
R7
220R
J3
U1.3
U1.4
From CPU
40106B
R6
220R
MIDI Out
The shield connector of the MIDI Out and MIDI thru connectors may be left unconnected or connected to ground
J1
Pin 2, no connection or optional 100nF capacitor to ground
The shield connector shall not have any DC path to the receivers ground
Tbot_MIDI_In

System Real-Time and Control message

System Real-Time and Control messages have a high nibble value of `1111` (bits 4-7). Bit 3 is a `0` for Common messages and a `1` for Real-Time messages. For example:

- Status byte `11110011` is a Common message for Song Select, it is followed by a single data byte `0sssssss`, where `sssssss` is the song number or sequence to be played.
- Status byte `11111000` is a Real-Time message for a timing clock. This status byte is sent 24 times per quarter note when synchronisation is required and there is no following data byte.

System Exclusive messages

SysEx (System Exclusive) messages expand the number of functions possible with MIDI and can be unique to a particular manufacturer that implements functionality specific to their instrument(s). The first Status byte is `1111000`, which is a System Common message to indicate a SysEx message follows. The SysEx status byte is immediately followed by data byte `0iiiiiii` where `iiiiiii` is a unique manufacturer's ID code issued under licence by **www.midi.org**

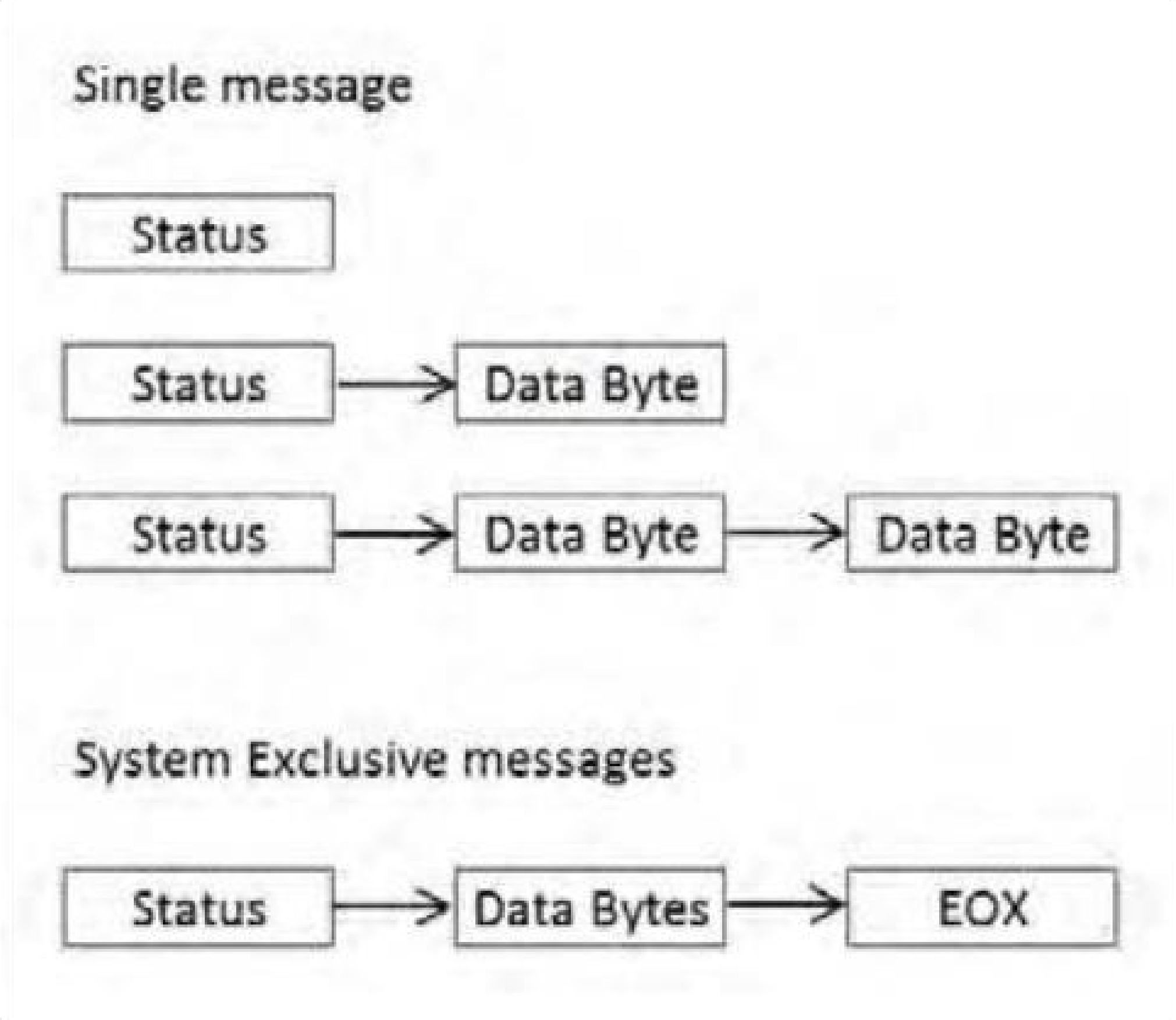


Fig.77. Bytes within a MIDI message type.

If the listening instrument recognises the manufacturer's ID code as its own, then it can listen to and act upon the rest of the message. After the manufacturer's ID code byte, any number of data bytes may follow until terminated by the System Common message 'End of Exclusive' (EOX) status byte `11110111`, or almost any other status byte. SysEx messages can be used to take control of an instrument – to dump patch memory, for example. You may have already noticed that the manufacturer's ID code is limited to 0-127, and naturally has long since been exceeded considering the number of manufacturers using MIDI. To get around this, if the manufacturer's ID code byte has a value of `0x00` (`0` in hex), then two further ID bytes are used to identify the manufacturer. Currently, the highest ID code assigned is `0x00 0x02 0x2D` so there is plenty of capacity for new manufacturers.

Despite the name, System Exclusive messages are not exclusive to manufacturers; some have been predefined by **www.midi.org** as universal for both Real-Time and Non-Real Time messages. Real-Time messages have a manufacturer's ID code of `0x7E` and Non-Real-Time use `0x7F`. If you are designing a MIDI instrument, then you may wish to consider responding to some of these universal messages, which can be found in Table 4: 'Defined Universal System Exclusive Messages', at: **www.midi.org**

Where do I go next?

While the above protocol is an overview of the capability of MIDI, you only need a fraction of this to implement MIDI in its most basic form, which is what we have done for the *MIDI Ultimate* synth. Should you have been encouraged to have a play with MIDI then there are various ways to get started. The **sparkfun.com** MIDI shield has the DIN sockets, opto-isolators and a few switches and pots to use as a MIDI controller. The shield plugs directly onto an Arduino Uno and the Uno has library functions to support sending and receiving MIDI messages. The *MIDI Ultimate* uses the same microprocessor as the Uno, which we will cover next. Developing and de-bugging systems using serial communications can be aided by a serial analyser; and fortunately, MIDI has this covered through the windows program MIDI-OX (visit **midiox.com** for a freeware download). Armed with a MIDI-to-USB converter, you can capture and analyse all incoming and outgoing MIDI messages, a very useful tool indeed. This leads us very nicely into how a MIDI interface was added to the *MIDI Ultimate* synth.

MIDI Ultimate MIDI Interface

Continuing on from where we left off in *Part 4* last month, we have one area of the main circuit board to populate, which is the MIDI interface. I noted that many DIY synths lacked a MIDI interface, which either limited the musical possibilities by the lack of a keyboard or required the purchase of a third-party MIDI-to-CV converter. From the previous parts of this series, you will be aware that analogue synths require a control voltage for the oscillators and filters, and a gate pulse for the envelope generators. The challenge is to take the MIDI messages described above from a MIDI keyboard and convert these into control voltages and gate pulses. The requirements for this MIDI interface are very basic because the *MIDI Ultimate* is monophonic and has no capability to electronically store patch arrangements, so we are primarily interested in when a note is pressed on a keyboard or is transmitted from a PC. To this we can add the expression controllers 'pitch bend' and 'modulation' wheels, commonly found even on the lowest-cost

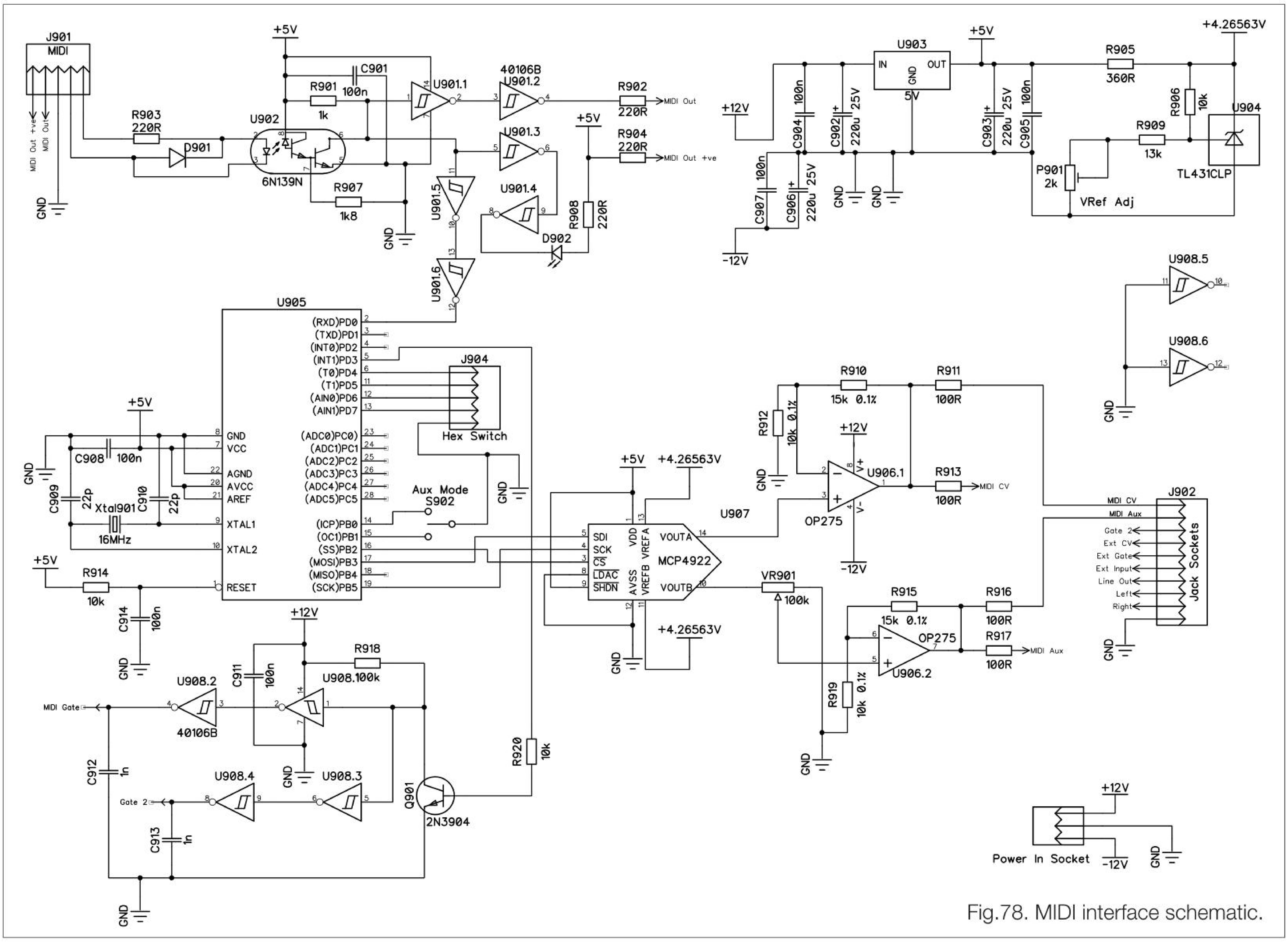

J901
MIDI
MIDI Out +ve
MIDI Out
GND
R903
220R
D901
U902
6N139N
+5V
R901
1k
C901
100n
R907
1k8
U901.1
40106B
U901.2
U901.3
U901.4
U901.5
U901.6
R902
220R
MIDI Out
R904
220R
MIDI Out +ve
R908
220R
D902
+12V
C907
100n
C906
220u 25V
C904
100n
C902
220u 25V
U903
IN
OUT
GND
5V
-12V
C903
220u 25V
C905
100n
R905
360R
+4.26563V
R906
10k
R909
13k
U904
TL431CLP
P901
2k
VRef Adj
U905
(RXD)PD0
(TXD)PD1
(INT0)PD2
(INT1)PD3
(T0)PD4
(T1)PD5
(AIN0)PD6
(AIN1)PD7
(ADC0)PC0
(ADC1)PC1
(ADC2)PC2
(ADC3)PC3
(ADC4)PC4
(ADC5)PC5
(ICP)PB0
(OC1)PB1
(SS)PB2
(MOSI)PB3
(MISO)PB4
(SCK)PB5
GND
VCC
AGND
AVCC
AREF
XTAL1
XTAL2
RESET
J904
Hex Switch
Aux Mode
S902
C908
100n
C909
22p
Xtal901
16MHz
C910
22p
R914
10k
C914
100n
U907
MCP4922
SDI
SCK
CS
LDAC
SHDN
VDD
VREFA
VOUTA
AVSS
VREFB
VOUTB
R910
15k 0.1%
R911
100R
R912
10k 0.1%
U906.1
OP275
R913
100R
MIDI CV
VR901
100k
R915
15k 0.1%
R916
100R
U906.2
R917
100R
MIDI Aux
R919
10k 0.1%
U908.5
U908.6
J902
Gate 2
Ext CV
Ext Gate
Ext Input
Line Out
Left
Right
Jack Sockets
U908.2
40106B
C911
100n
U908.100k
R918
MIDI Gate
C912
1n
U908.4
U908.3
Gate 2
C913
1n
R920
10k
Q901
2N3904
Power In Socket

Fig.78. MIDI interface schematic.

MIDI keyboards. This article covers far more theory than previous parts, but if you want to get on with your *MIDI Ultimate* build first then jump to *Assembly*.

Design solution

Rather than just state how the MIDI interface works, as with previous circuit descriptions, I will go into the design in a bit more depth to aid those who wish to develop their own MIDI designs or to 'hack' the *MIDI Ultimate*. You may need to refer to the general introduction to MIDI in the first part of this article if you are unsure of any terminology used.

Let's start with 'Note On'. We will need to be able to tell which key note has been pressed and convert that into a control voltage. At the same time, create a gate pulse for as long as the key remains pressed. The note number transmitted with a Note On message can have a value within the range 0-127. Not many converters can or need to be able to operate with all 128 notes, some are limited to four or five octaves, some offer a full eight octaves, as on a piano. This design can quite happily work with all 128 notes or over 10 octaves without the need for high- resolution digital-to-analogue converters (DAC).

In *Part 2*, the relationship between the frequencies of each note on a keyboard was described (see also Table 2). That exponential relationship is taken care of in the VCOs, so our MIDI interface needs to create the 1V/octave linear progression. The octave number used in the following text is shown in the superscript after the note name. So if we start at MIDI note 0, which is note C^{-1} on a keyboard (a piano starts at key A^0), the voltage should be 0V. C^0 is 1V, C^1 is 2V and so on until MIDI note 120 which is C^9, represented by 10V. For completeness, MIDI note 127 (G^9) would be 10.5833V. Any rounding errors in the DAC or a lack of resolution would mean the DAC output is not the desired voltage and thus de-tune the VCOs. One approach is to use 16-bit DACs, but in this application a 12-bit DAC can perform equally well, even though it only has 2^{12} steps = 4096, which at best means a step size of 2.604mV. Each MIDI note is 83.333mV (1V/12 semitones) higher than the previous one, so a 2.604mV 12-bit DAC resolution may initially seem as not so desirable. However, 2.604167mV × 32 = 83.33333mV, which is *exactly* the step voltage we need for each semitone. The number 32 is used in the software, which is explained later.

Key C^{-1} has a frequency of 8.175799Hz, C^0 is 16.351598Hz, C^4 is 261.625565Hz and so on, maintaining the principle that each note is twice the frequency of the same note of one octave lower. We can predict the frequency of a note given the DAC voltage as: $2^{\text{DAC Voltage}} \times 8.175798916$, for example C^4 gives $2^5\text{V} \times 8.175798916 = 261.6255653\text{Hz}$. Looking at the calculated frequency errors for any MIDI note (see Table 6), the theoretical error is less than 1μHz. In practice, the DACs have an error with their conversion process that may be as high as ±2 bits, but after even allowing for that, it has minimal impact for the lower notes, at the highest piano note of C^8, it could cause a frequency error of 15Hz. The piano tuner would be very skilful to detect an error of 15Hz on a note frequency of 4186Hz! For the lowest piano note (A^0) the error is less than 0.1Hz.

MIDI interface circuit description

The schematic in Fig.78 is the complete MIDI interface. From what we covered earlier in the MIDI electrical

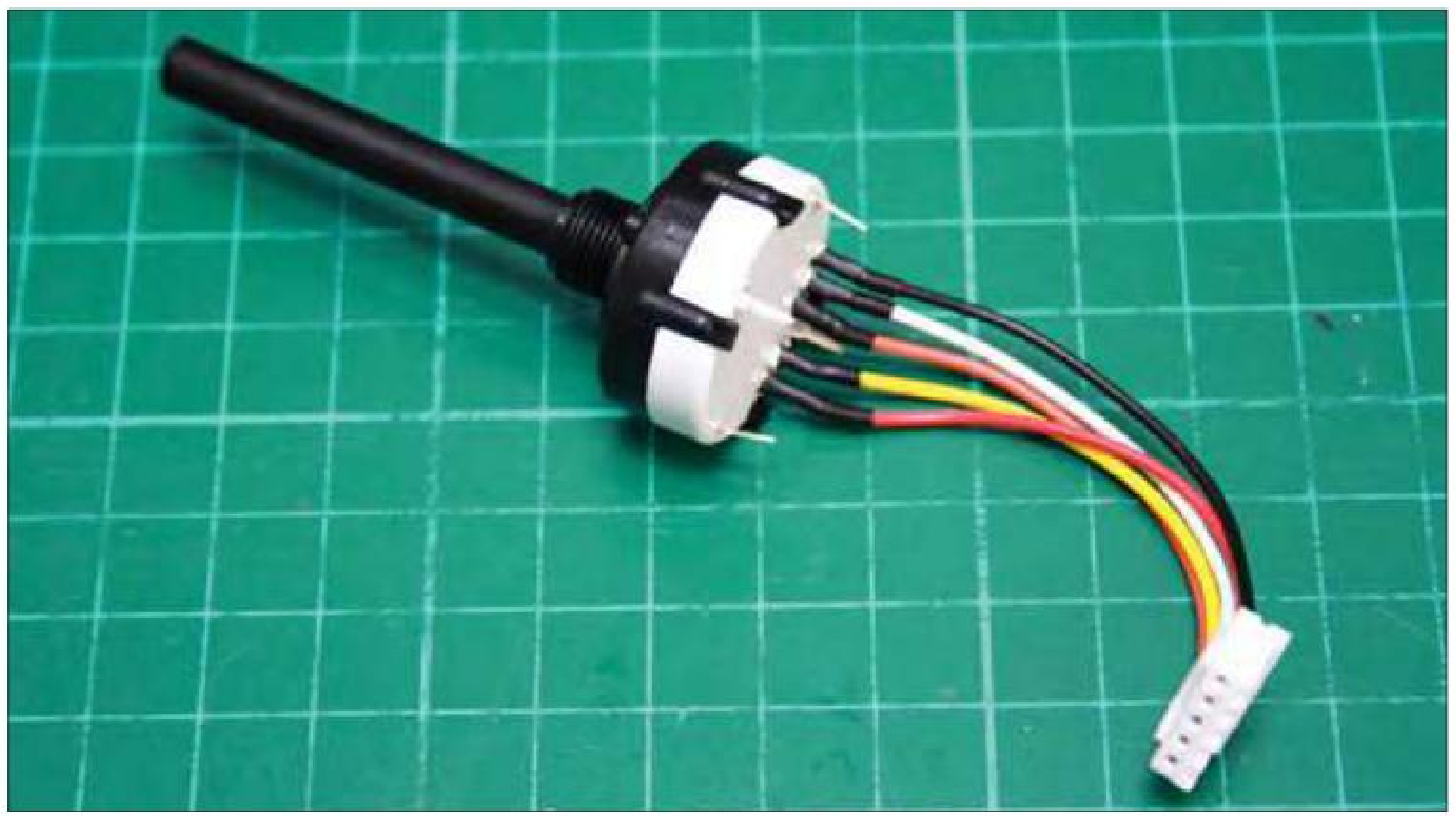


Fig.79. JST lead soldered to the Hex switch, black (A), red (B), white (C), yellow (D), orange (E).

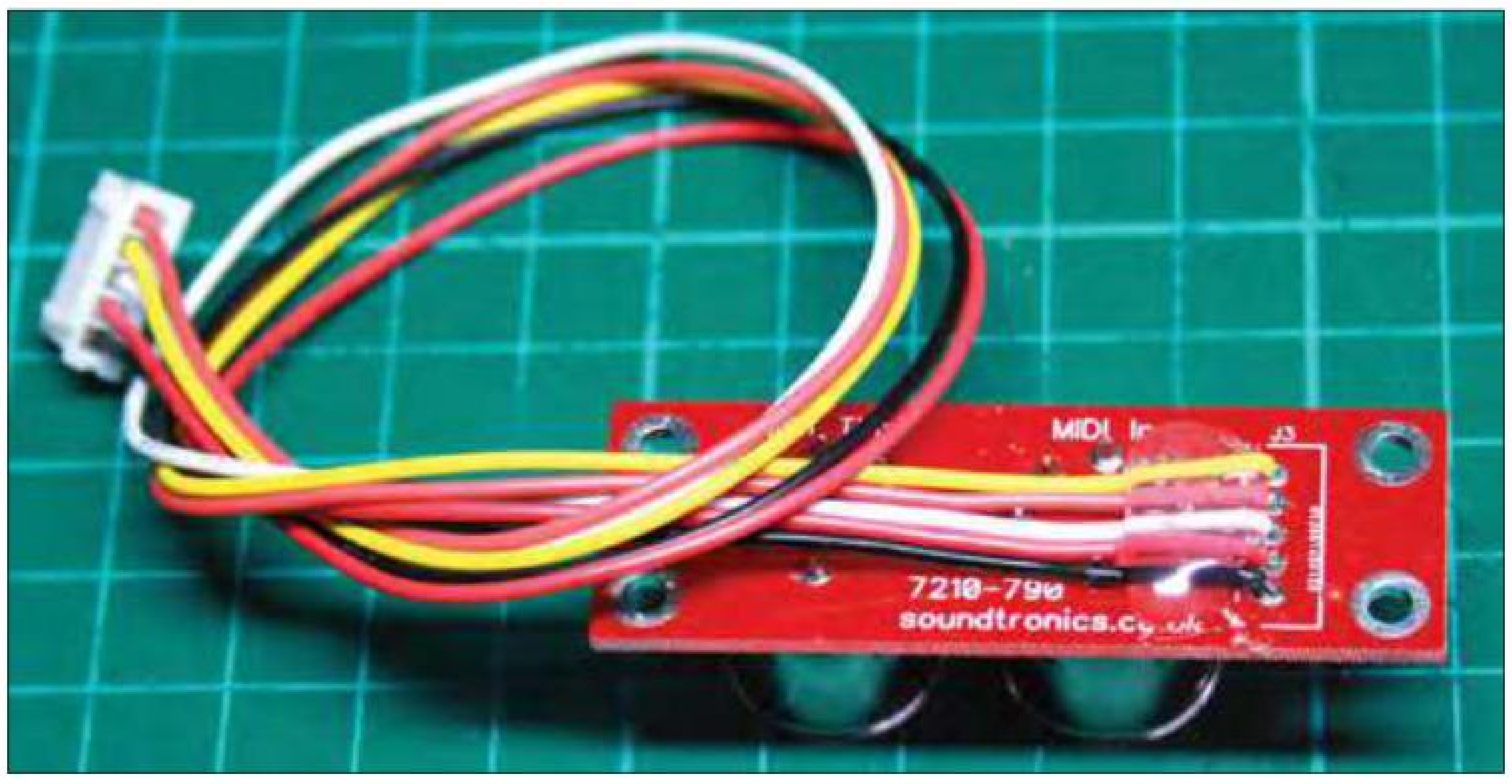


Fig.80. Wiring to the MIDI sockets breakout board. From this view, top down, the wiring colours are yellow, orange, white, red, black. Use of hot glue is recommended as a cable strain relief.

specification, U902 should be self-explanatory as the opto-isolator for the MIDI In signal. Schmitt inverting buffers are used for the MIDI Thru, U901.1 and U901.2, U901.3 and U901.4 to drive an LED to show MIDI messages are being received, U901.5 and U901.6 are used to ensure a clean pulse train into the CPU. U903 provides the dedicated 5V supply for the CPU (U905) and the DAC (U907). A DAC reference voltage of nominally 4.26563V is provided by U904, which is a precision programmable voltage reference (TL431CLP) that can be fine adjusted by P901.

The DAC is actually a dual channel MCP4922; one channel for the main control voltage and the other for the expression controller selected using switch S902. The DAC communicates with the CPU (U905) using an SPI communications interface. The DAC outputs are buffered by U906, which also provides a voltage gain of 2.5 to achieve the required 1V/octave. U906 was selected because it has a low offset and so does not require any offset trimming – I use 0.1% tolerance resistors to determine the gain, so adjustment is not required here either.

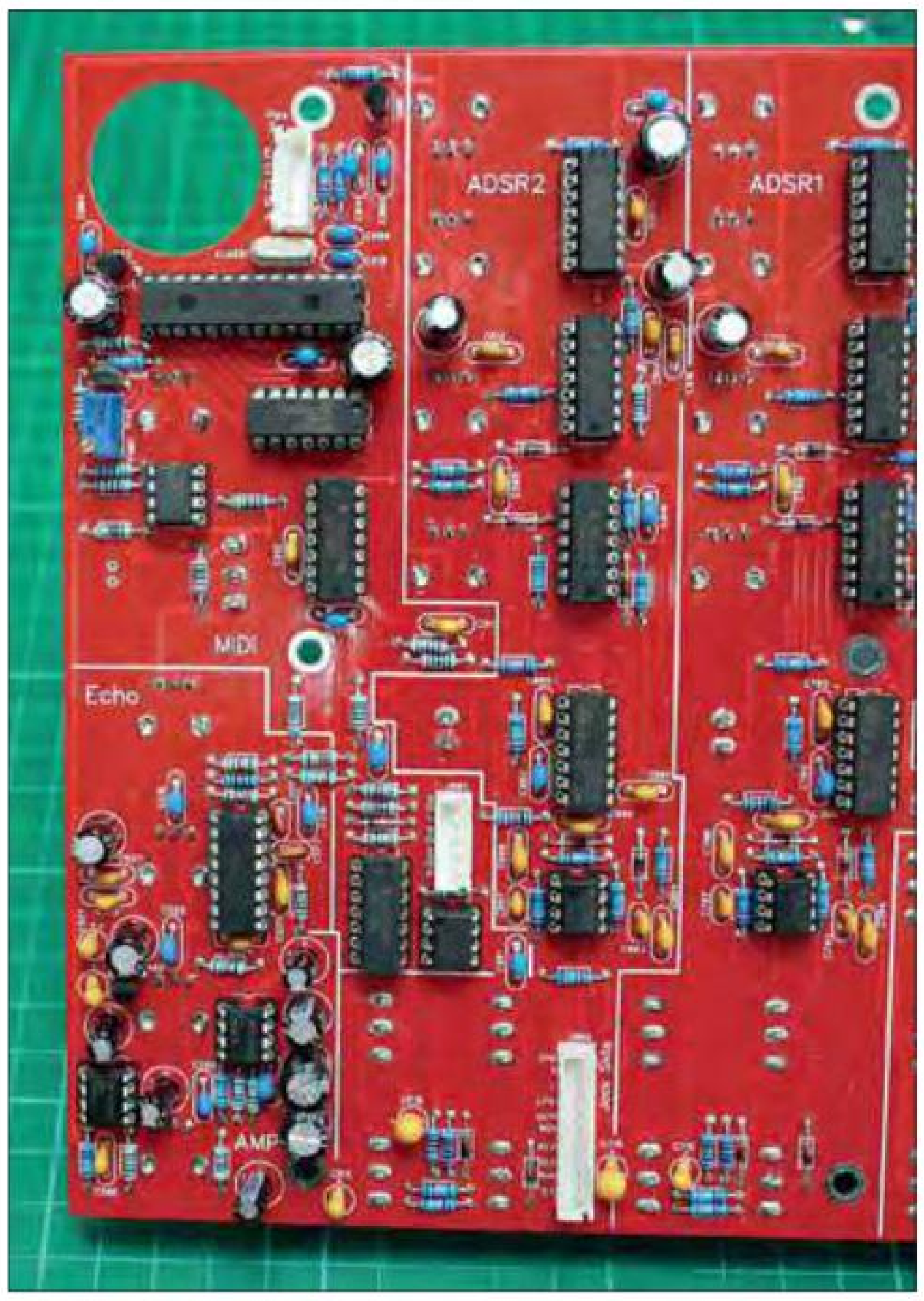


Fig.81. Now that the MIDI section (top left) is populated, the main PCB is complete.

The gate signal from the CPU is buffered by U908 to produce two gates, one for internal use and the other for the external jack socket. Q1 is a voltage-level shifter, so the 5V from the CPU can drive the 12V gate pulse.

The final part of the circuit is the CPU, this is an ATMEGA328P-PU which is the same processor used in the Arduino UNO and is supplied pre-programmed in the Soundtronics kits but note that the code below allows users to program their own if they prefer. The CPU requires minimal components to function: a 16MHz crystal Xtal901, R914/C915 to ensure the CPU is held at reset during power up and that's it. The last component to mention is J904; a connector used to connect to a binary coded 16-way (hex) rotary switch that is used to select which MIDI channel the synth will listen to.

Assembly

This is now the last stage in populating components on the main PCB (Fig.81) and follows along the same methodology as used in previous articles in the series, but note the following:

- Using a 5-way JST lead cut to 60mm in length, solder its wires to the hex switch, and use sleeving or heatshrink over the switch pins (Fig.79).
- Mount the two MIDI sockets to the MIDI socket breakout board. Using the second 5-way JST lead (but do not cut off any the lead length), solder its wires to breakout board (Fig.80).
- A few MIDI interface components (R901, C906 and C907) have been positioned just outside the demarcated MIDI area, but at this stage, they will be easy to spot.

Calibration

Connect up the board as shown in Fig.82. P901 is used to set the voltage references for the two DACs. Initially, set it to 4.26V by measuring the voltage on pin 13 of U907. Now take a MIDI keyboard set to MIDI channel 1 and plug a MIDI lead between the keyboard and the synth MIDI in socket. I used a low-cost two-octave keyboard with which you can transpose the keyboard ±3 octaves using its buttons.

While monitoring the MIDI CV output using the grey wire from J902, press key C4, which should give close to 5.00V. If you have the wrong C key, go up or down octaves until you get around 5V out. The more accurate you can measure this, the better the results. Adjust P901 until the output is exactly 5.00V or 5.000V if your meter has enough (real) resolution. It is more important to be accurate for higher notes than lower notes, hence the setting at 5V. (Table 7 demonstrates that larger errors at the lower end have less impact on the actual frequency.)

To check the remaining MIDI functions, measure the voltage on the violet wire from J902, which should be at about 12V when a keyboard key is pressed and 0V when released. The auxiliary MIDI signal can be checked on the grey wire of J902, but make sure VR901 pot is fully clockwise. The function of the auxiliary MIDI signal depends on the setting of S902, so you should check all three modes. The pitch bend wheel (1.5V ±1.3V) and modulation wheel (0-10.26V) are self-explanatory; the velocity voltage varies depending on how quickly you strike the keys (~0-10V). Next month, we will calibrate the VCOs now that we have an accurate voltage source in the MIDI CV signal that we can use to create 1V increments.

Software

The software (Listing 1) is very straightforward thanks to three library functions that do the hard work. These are **SPI.h** for the interface to the DAC (U907), **DAC_MCP49xx.h** for the DAC itself and **MIDI.h**. These libraries will need to be in the Arduino libraries folder to compile.

Jumping straight to the main program in the `void loop()`, there are just four lines of code.

Line ❶ – The first is a function call to find out what MIDI channel the rotary switch connected to J904 is set to. The switch is a Lorlin BCK1002, which has a 4-bit code (Table 8) representing the switch position.

The function `CalcMidiCh()` reads the status of each of the switch lines (A to D) and returns a number of 1-16 in variable `MidiCh`.

Line ❷ – A function call to determine what position the auxiliary switch is in. Function ModeSwitch reads the two switch lines to determine whether modulation, velocity or pitch bend has been selected.

Line ❸ – Instructs the MIDI function to only listen to messages on the channel defined in variable `MidiCh`.

Line ❹ – Instructs the MIDI function to check for any messages. In the `void setup()` section of the program – four message types were specified. Should any messages of this type be received, then the corresponding function is called. The four message types are:

```
MIDI.setHandleNoteOn(handleNoteOn)
MIDI.setHandleNoteOff(handleNoteOff)
MIDI.setHandlePitchBend(pitchbend)
MIDI.setHandleControlChange(control)
```

Table 7: Actual results of the authors MIDI control voltage.

Required voltage (V)	Actual voltage (V)	Required frequency (Hz)	Frequency error (Hz)
8.000	8.000	2093.00	+1.4
7.000	6.998	1046.50	–1.5
6.000	6.004	523.25	+1.45
5.000	5.000	261.63	0
4.000	4.0033	130.81	+0.3
3.000	3.0027	65.41	+0.11
2.000	1.9945	32.70	–0.12
1.000	0.9935	16.35	–0.07
0.000	0.0018	8.176	+0.01

A function call to `handleNoteOn` occurs when a Note On message on the selected MIDI channel is received. This function turns off the Gate signal to ensure ADSR envelope generators are re-triggered. The MIDI note number is stored in the variable `LastNote` because it is needed for Note Off later. In the *Design solution* section above, the number 32 was discussed. Each MIDI note number is multiplied by 32 to give the value sent to the DAC for determining the VCO frequency. So, MIDI note 0 gives a DAC value of 0, 1 gives 32, 2 gives 64, 3 gives 96 all the way through to 127, which gives 4064. This means we make a near full use of the DAC's maximum count of 4095. Included in the Note On message is the key velocity, which basically means the higher the number (0-127), the faster the key was struck. If the auxiliary switch is selected to velocity, then the second DAC is assigned a value 32× the velocity, giving a fully 0 to >10V output. Finally, the gate output is made active (low).

A function call to `handleNoteOff` occurs every time a key is released. This can be a problem for monophonic synths in that your playing style may result in more than one key being depressed at a time due to overlapping. We do not want the last note played to be turned off just because a previous note was released late. This is managed by only removing the gate signal if the Note Off number matches the last note played using variable `LastNote`. Over time, manufacturers have implemented various approaches to how gate pulses and note priorities are managed. Possible gate modes include Single, Multiple and Multiple + Release. Priorities include Only Last, Any Last, Low and High. The *MIDI*

Table 6: Extract of Voltage frequency relationship of DAC table

Piano Key	Midi Note	Key Name	Note Number Relative to A4 'n'	Midi Note × 32	12-bit DAC Voltage	Op-Amp Gain 2.5	Actual Frequency (Hz)	Desired Freq 'Fn' (Hz)	Error in Freq Hz
37	57	A3	–12	1824	1.90000	4.75000	220.00000001	220.00000000	0.00000001
38	58	A#3	–11	1856	1.93333	4.83333	233.08188077	233.08188076	0.00000001
39	59	B3	–10	1888	1.96667	4.91667	246.94165064	246.94165063	0.00000001
40	60	C4	–9	1920	2.00000	5.00000	261.62556531	261.62556530	0.00000001
41	61	C#4	–8	1952	2.03333	5.08333	277.18263099	277.18263098	0.00000001
42	62	D4	–7	1984	2.06667	5.16667	293.66476793	293.66476792	0.00000001
43	63	D#4	–6	2016	2.10000	5.25000	311.12698374	311.12698372	0.00000001
44	64	E4	–5	2048	2.13333	5.33333	329.62755693	329.62755691	0.00000001
45	65	F4	–4	2080	2.16667	5.41667	349.22823145	349.22823143	0.00000001
46	66	F#4	–3	2112	2.20000	5.50000	369.99442273	369.99442271	0.00000002
47	67	G4	–2	2144	2.23333	5.58333	391.99543600	391.99543598	0.00000002
48	68	G#4	–1	2176	2.26667	5.66667	415.30469760	415.30469758	0.00000002
49	69	A4	0	2208	2.30000	5.75000	440.00000002	440.00000000	0.00000002
50	70	A#4	1	2240	2.33333	5.83333	466.16376154	466.16376152	0.00000002
51	71	B4	2	2272	2.36667	5.91667	493.88330128	493.88330126	0.00000002
52	72	C5	3	2304	2.40000	6.00000	523.25113062	523.25113060	0.00000002
53	73	C#5	4	2336	2.43333	6.08333	554.36526198	554.36526195	0.00000002

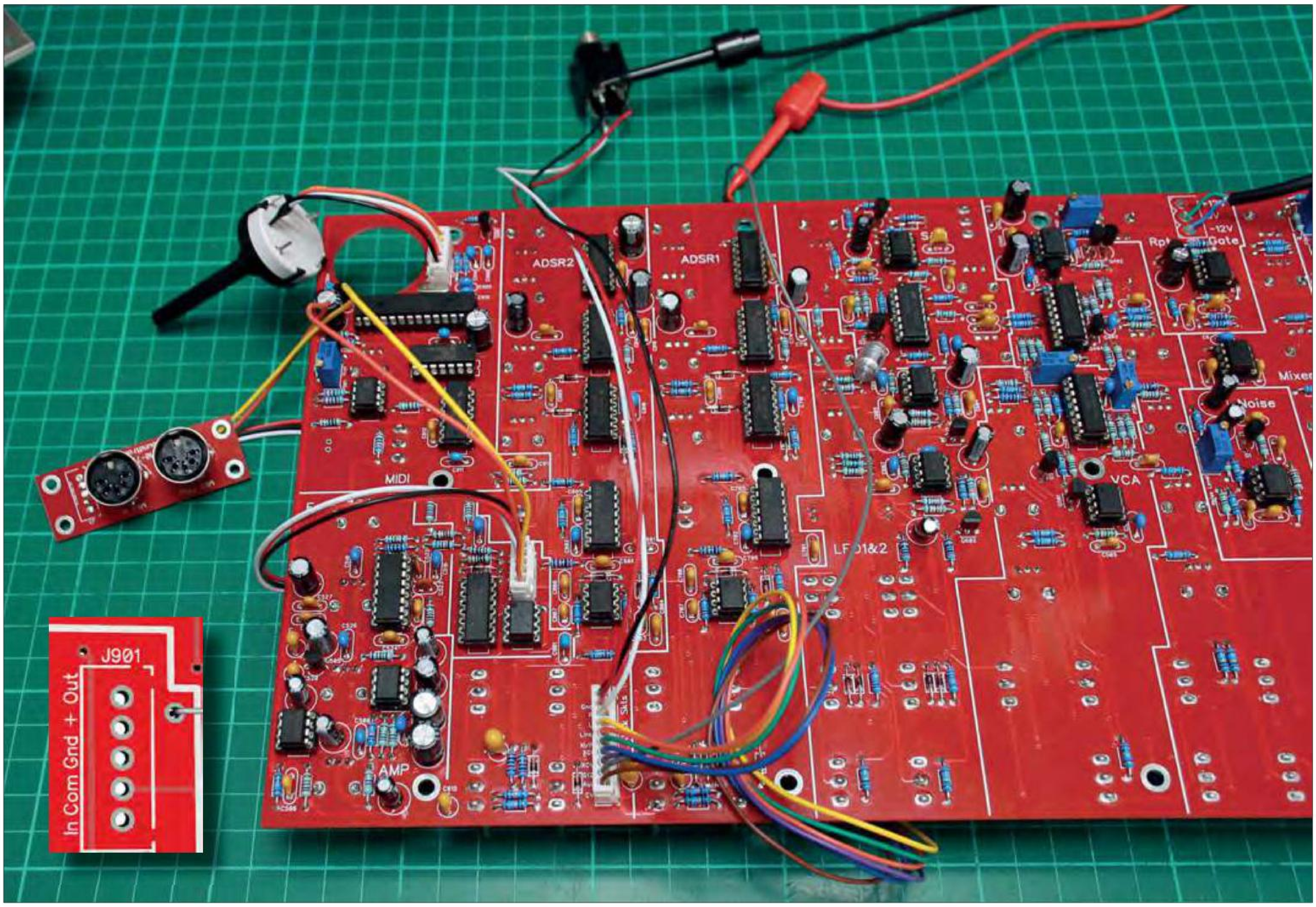


Fig.82. MIDI section ready for testing, the MIDI CV output can be measured on the grey wire from J902. (**Erratum:** J901 silk screen legend has '+' and 'Out' swapped, please follow the colour sequence in Fig 80. A corrected silk screen detail is shown above left.)

Table 8: Truth table for MIDI channel selector switch

Position	A	B	C	D
1	*			
2		*		
3	*	*		
4			*	
5	*		*	
6		*	*	
7	*	*	*	
8				*
9	*			*
10		*		*
11	*	*		*
12			*	*
13	*		*	*
14		*	*	*
15	*	*	*	*
16				

Ultimate uses Multiple and Only Last, which mean:

Multiple – a new Gate signal is produced each time a new key is pressed, even if other keys are being pressed simultaneously. This new Gate is generated by turning the Gate off for a very short period of time, then back on. New Gates will retrigger envelope generators as each key is pressed, regardless of the number of keys being held down.

Only Last – the last note played determines the voltage at the Pitch output. This provides a very intuitive keyboard response. Regardless of the direction that your note sequence is played, the most recent note is produced at the Pitch output.

A function call to `PitchBend` occurs each time that the value of the pitch bend wheel changes. The *MIDI Ultimate* has only a simple implementation of pitch bend, usually the pitch bend has a 0V output when it is in the spring return to centre position. As we have complete control of the starting frequency of the VCOs, we can set them to compensate for the offset. The output to the DAC is 1.5V ±1.3V (±16 semitones) as the wheel is moved up and down respectively.

A function call to `control` occurs each time the modulation wheel value changes. The *MIDI Ultimate* does not have an actual modulation function, but the modulation wheel can be used expressively in a different way to the pitch bend, as it is not spring return to centre. This control message includes two data bytes, the first of which is the controller number. In our case, the controller number is 1, which is for the modulation wheel. You can change this number to 4 for a foot controller, or to any one of the 120 possible controllers. The second data byte is the value of the controller in the range 0-127.

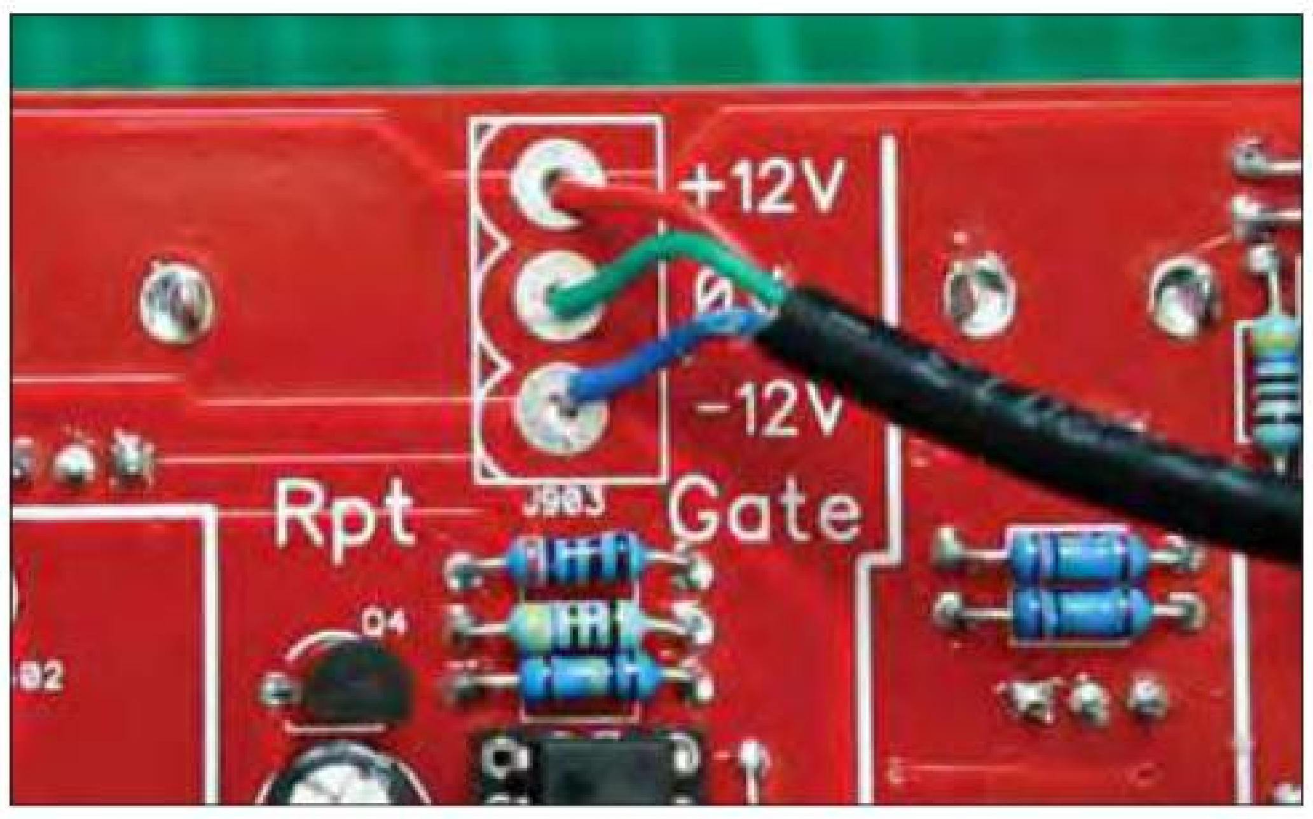


Fig.83. Power wiring to main PCB, do not use a power connector if housing in the Soundtronics hardwood case – details next month.

```
#include <MIDI.h>
#include <SPI.h>
#include <DAC_MCP49xx.h>
// https://github.com/FortySevenEffects/arduino_midi_library
// https://github.com/exscape/electronics/tree/master/
Arduino/Libraries/DAC_MCP49xx

// This work is licensed by Paul Cooper of Soundtronics Ltd
under the Creative Commons Attribution-NonCommercial-ShareAlike
4.0 International License. To view a copy of this license,
visit http://creativecommons.org/licenses/by-nc-sa/4.0/
MIDI_CREATE_DEFAULT_INSTANCE();

//Assign Pins with user name
const byte Gate=3;
const byte HexA=4;
const byte HexB=5;
const byte HexC=6;
const byte HexD=7;
const byte Vel=8;
const byte Bend=9;
const byte SS_PIN = 10; // The Arduino pin used for the
slave select / chip select

byte MidiCh=1; //Set default Midi channel to channel 1
byte Mode = 0; // 0 = Mod, 1 = Velocity, 2 = Bend
byte LastNote;

// Set up the DAC.
    // First argument: DAC model (MCP4902, MCP4912, MCP4922)
    // Second argument: SS pin (10 is preferred)
    // (The third argument, the LDAC pin, can be left out if
not used)
    DAC_MCP49xx dac(DAC_MCP49xx::MCP4922, SS_PIN);


void setup()
{
    //Configure Pins
    pinMode(HexA, INPUT_PULLUP);
    pinMode(HexB, INPUT_PULLUP);
    pinMode(HexC, INPUT_PULLUP);
    pinMode(HexD, INPUT_PULLUP);
    pinMode(Vel, INPUT_PULLUP);
    pinMode(Bend, INPUT_PULLUP);
    pinMode(Gate, OUTPUT);

    //Set-up MIDI function calls
    MIDI.setHandleNoteOn(handleNoteOn);
    MIDI.setHandleNoteOff(handleNoteOff);
    MIDI.setHandlePitchBend(pitchbend);
    MIDI.setHandleControlChange(control);
    // Initiate MIDI communications on channel 0
    MIDI.begin(1);

    digitalWrite (Gate, HIGH); //Set gate output high, this
gets inverted by the voltage level shifter transistor
}
```

```
void loop()
{
❶      MidiCh = CalcMidiCh(); //Function call to calculate Midi
Channel selected
❷      Mode = ModeSwitch(); // Function call to detect mode
switch position
❸      MIDI.setInputChannel(MidiCh); //Set Midi channel to that
selected by channel hex switch
❹      MIDI.read();

}
```

```
//This function calculates the selected Midi channel from
the hex switch type Lorlin BCK1002 in the range 1-16
    int CalcMidiCh(){
    int result;
    bool A;
    bool B;
    bool C;
    bool D;
    A = !digitalRead (HexA);
    B = !digitalRead (HexB);
    C = !digitalRead (HexC);
    D = !digitalRead (HexD);
    result = A | (B << 1) | (C << 2) | (D << 3);
    if (result == 0){
      result = 16;
    }
    return result;
}

int ModeSwitch(){
    int result;
    int A;
    int B;
    A = !digitalRead (Vel);
    B = !digitalRead (Bend);
    result = A | (B << 1);
    return result;
}

void handleNoteOn(byte channel, byte pitch, byte velocity)
{
  //Do the following when a note is pressed - Note priority is
  digitalWrite(Gate, HIGH);
  LastNote = pitch;
  delay(10);
  dac.outputA(pitch*32);//128 MIDI notes which when multiplied
by 32 gives a dac range of 0 - 4064 in 32 bit steps
  if (Mode == 1){   //Velocity mode
    dac.outputB(velocity*32);
  }
  digitalWrite(Gate, LOW);
}

void handleNoteOff(byte channel, byte pitch, byte velocity)
{
  if (pitch == LastNote){
    digitalWrite(Gate, HIGH);//Turn off gate only when the
last key pressed is released
  }
}

void pitchbend(byte channel, int bend)
{
  delayMicroseconds(1);
  bend=((bend/16)-3500);
  if (Mode == 2){
    dac.outputB(bend);
  }
}

void control(byte channel, byte number, byte value)
{
  if ((number == 1) & (Mode == 0)){ // Modulation wheel
  delayMicroseconds(1);
  dac.outputB(value*31);
  }
}
```

Listing 1. *MIDI Ultimate* programme using the Arduino IDE

MIDI Ultimate code

This code is available for download from the *PE* website.

Hacking the code

You can download the code from the PE website. Copy and paste the above code into your Arduino IDE, ensure the library functions have been installed and ensure the code verifies. The CPU U905 can be removed and plugged into an Arduino Uno but you will need the type of UNO where the ATMEGA328-PU is a socketed DIL chip, not the surface mount type. From that point, you can freely modify the code, you could add a sequencer to send a string of notes on power up, or re-use the mode switch in some way. You could also experiment with the gate and note priority logic; ie, low-note priority for the Moog style.

Important – power socket

If you intend to house the *MIDI Ultimate* in the Soundtronics hardwood case, do not fit a power socket on the main PCB, simply solder wires/cable direct to the PCB.

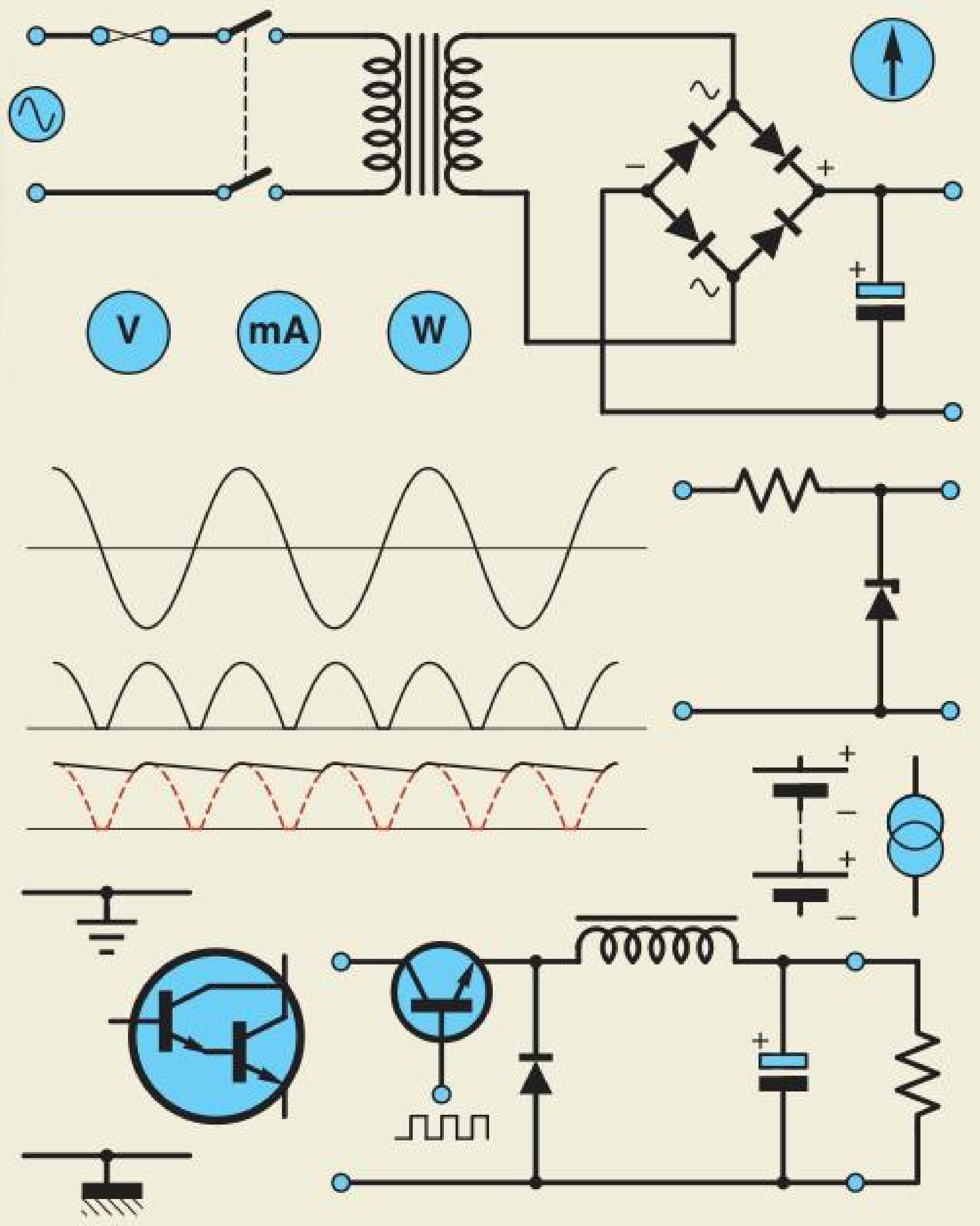
Next month

There will be more on cases in *Part 6* next month, which will cover building the case, final assembly, connections to the jack sockets and the synth's final calibration. Plus, I'll provide patch suggestions and ideas on developing your interest in analogue synths.

Teach-In 2019

Powering Electronics

Part 7: Heatsinks and negative outputs



by Mike Tooley

Your project is finished and ready to go, but the job isn't done until you've found an appropriate source of power. This could be as simple as choosing a suitably rated mains adapter or as complex as designing a switched-mode power supply with multiple outputs and battery backup. Our latest series – *Teach-In 2019* – is here to help, and will provide you with insight into all aspects of powering your electronic projects and designs.

Last month, in Part 6, we introduced switched-mode power supplies (SMPS) and explained the basic principles of buck (step-down) and boost (step-up) switching regulators. This month, we will look at two topics.

First, some simple arrangements for producing negative outputs from a positive DC supply.

Second, all electronic circuits produce heat when they operate, and this can be esprecially significant in power supplies. Managing the safe dissipation of heat can be a crucial factor in determining the reliability of a power supply, so we include some important advice on heat dissipation as well as selecting and using heatsinks.

Our *Practical Project* is a low-current negative supply converter that will typically produce an output of –5V at 15mA from a nominal +12V input.

Negative outputs from a positive DC input

It is occasionally necessary to provide a negative supply rail when only a positive DC input is available. With an AC supply this isn't a problem, since multiple positive and negative outputs can easily be derived from simple rectifier/regulator configurations, as shown in Fig.7.1.

When only DC input is available it is necessary to use switching techniques to generate a negative output. Where relatively high currents are required an SMPS can be used with a transformer having multiple secondary windings (see Fig.4.5 on page 45 of *EPE*, March 2019). However, where only low-currents are needed (<100mA) a less complex solution can be based on a circuit known as a 'charge pump'.

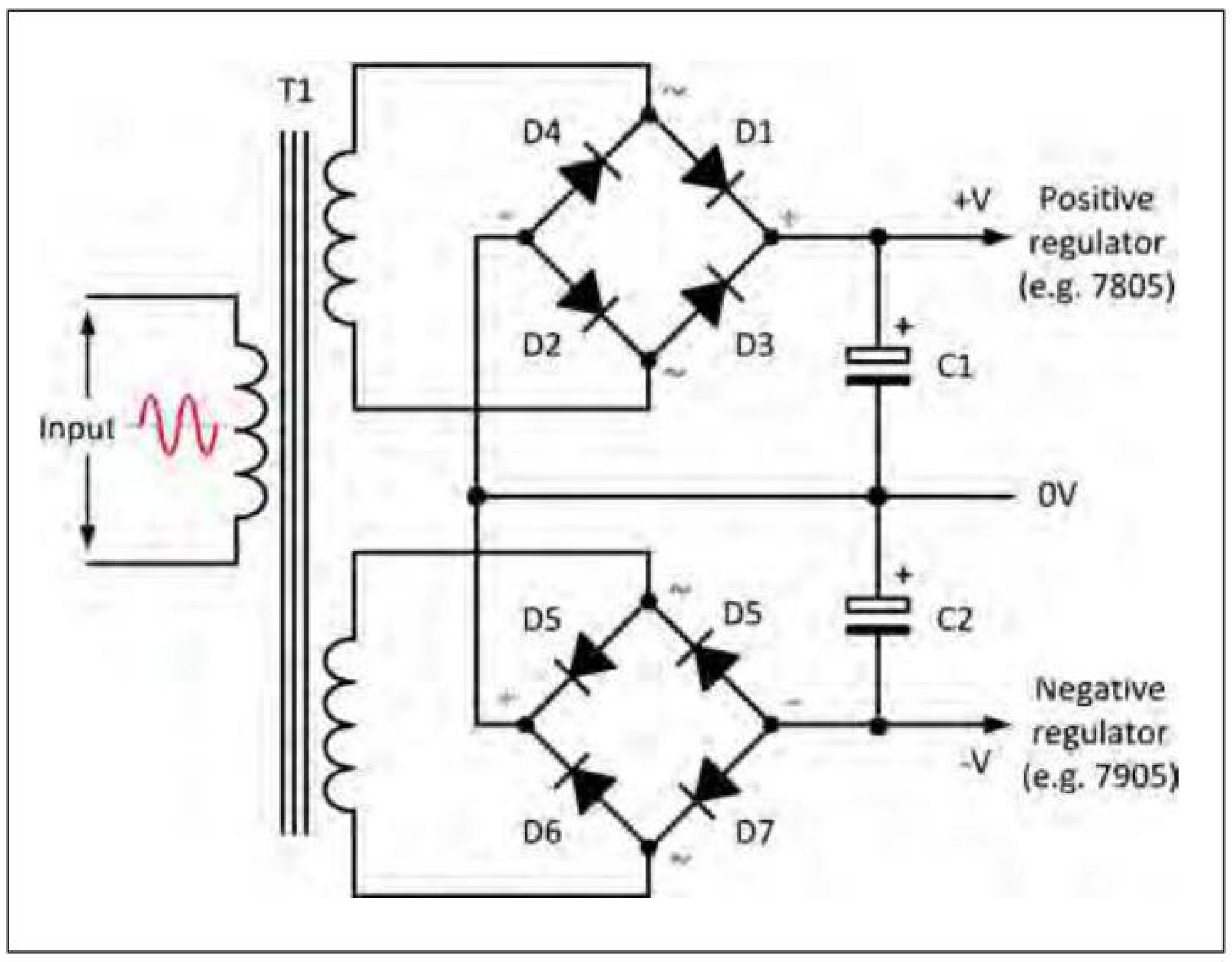
Charge pumps

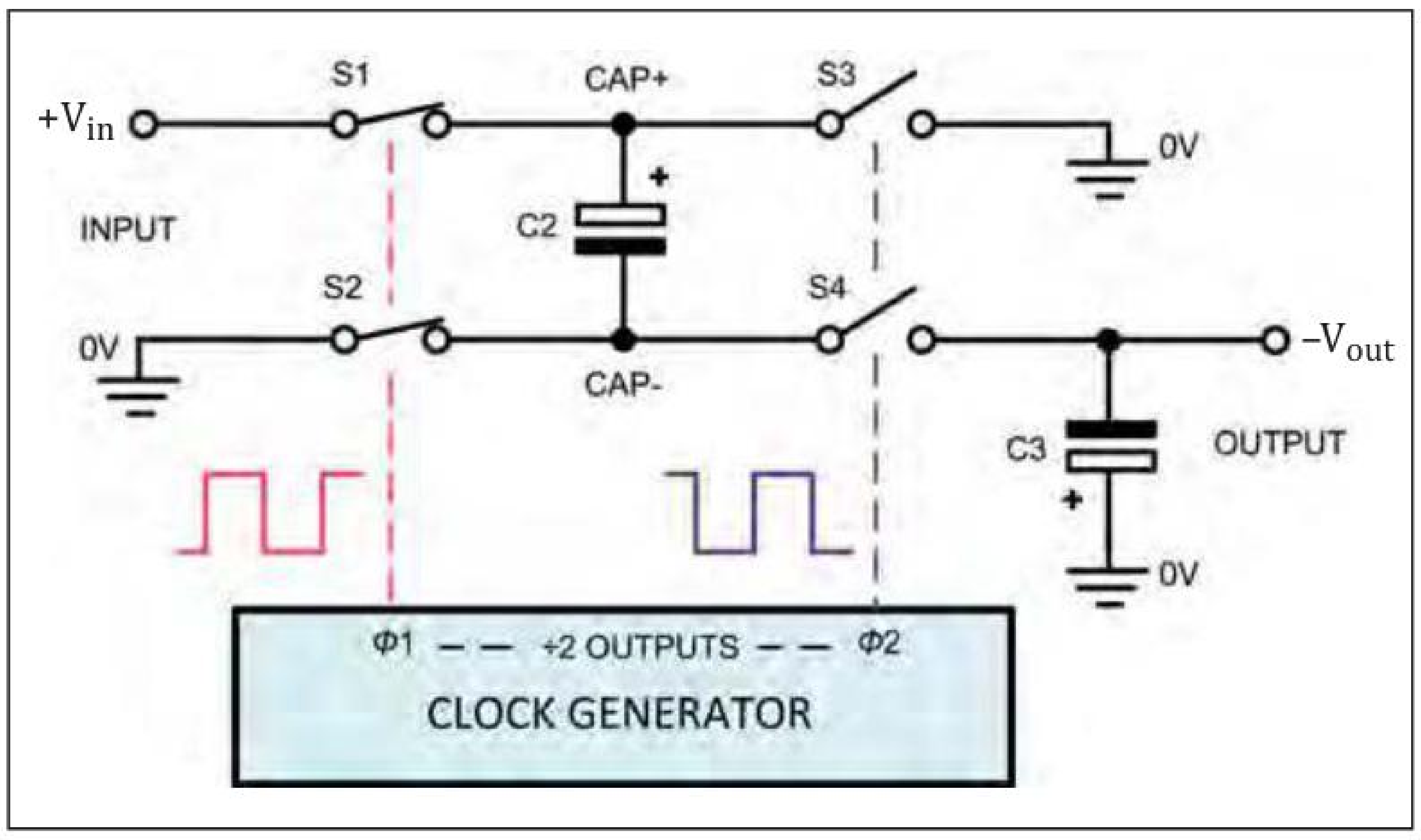
The principle of the charge pump is shown in Fig.7.2. This arrangement is based on the Analog Devices ADM660 and ADM8660 chips (the ADM660 can operate both as a voltage inverter *and* a voltage doubler).

When operating as voltage inverters, the ADM660/8660 charge pumps will convert an input in the range +1.5V to +7V to a corresponding negative output in the range –1.5V to –7V. Since only two external capacitors are required for the charge pump and maximum efficiencies of around 90% are achievable at load currents of up to 50mA, this inverting scheme is ideal for generating a negative rail in a low-voltage single-supply system.

A frequency control input (FC) selects charge-pump operation at either 25kHz or 120kHz. For 25kHz operation, the charge pump capacitors (C2 and C3 in Fig.7.2) should be 10µF low-ESR miniature radial lead electrolytic capacitors (see last month); while for 120kHz the values can be reduced to 2.2µF (again, low-ESR components should be used).

The internal clock generator in the ADM660/8660 chips provides two anti-phase outputs at half the fundamental clock frequency. These two outputs control pairs of switches (S1/S2 and S3/S4). Note that when S1/S2 are closed, S3/


T1
D4
D1
D2
D3
+V
Positive regulator (e.g. 7805)
C1
Input
0V
D5
D5
C2
Negative regulator (e.g. 7905)
-V
D6
D7



(left) Fig.7.1. A typical AC input supply with separate positive and negative outputs. (above) Fig.7.2. Principle of a charge-pump negative voltage converter.

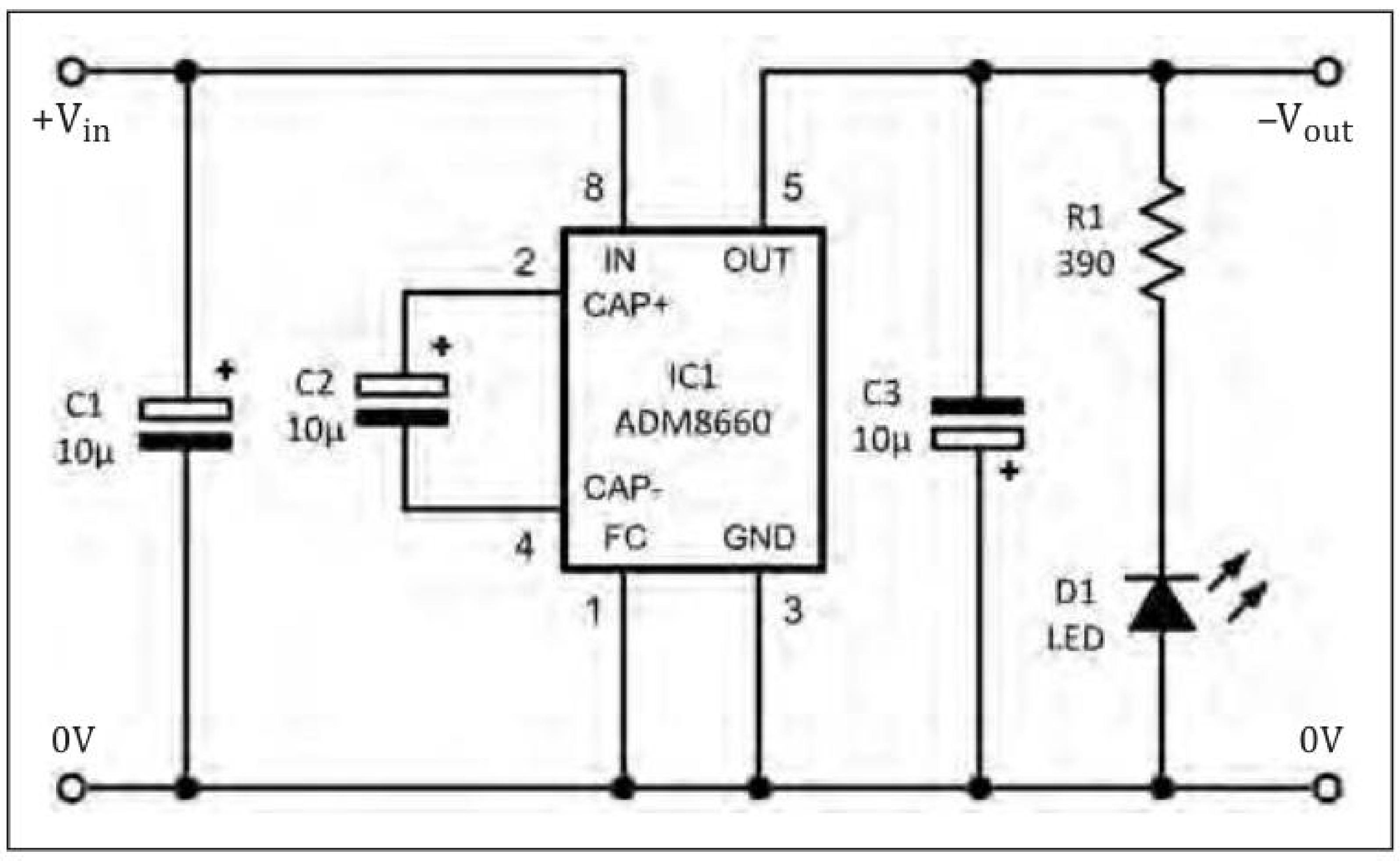


Fig.7.3. Voltage inverter based on an ADM8660 charge-pump; capable of delivering output currents of up to 100mA.

S4 are open, and vice versa. When S1/S2 are closed and S3/S4 are open, C2 is charged to the positive input voltage. When S1/S2 are open and S3/S4 closed, the charge in C2 is transferred to C3. Since the positive electrode of C3 is grounded (ie, 0V) a negative output will appear at its negative electrode. Thus, the input voltage is effectively inverted (ie, the output will have the same magnitude as the input, but it will be of opposite polarity).

Fig.7.3 shows a typical circuit arrangement based on the ADM8660 with the addition of a noise-reducing input capacitor (C1) and an output voltage indicator (R1 and D1). With a –5V input this circuit will produce a no-load output of –5V. With a load of 50mA the output will fall to about –4.5V, and with maximum load of 100mA the output will be just less than –3.9V.

Our *Practical Project* this month features an alternative approach to low-current negative supplies in the form of a 555 converter, similar to the voltage doubler last month, but producing a negative output voltage rather than a positive output.

Selecting and fitting heatsinks

The electronic equipment designer is often faced with the task of developing power supplies that will operate within a range of temperatures that guarantee safe and reliable operation. This usually requires the application of some simple mathematics to the design of heatsinks and heat dissipators. The algebraic symbols most commonly used in heatsink calculations are summarised in Table 7.1

Derating and maximum junction temperature

When a semi-conductor device such as a diode, transistor or integrated circuit is operating, heat will be generated. This heat will produce an increase in temperature, which, if excessive, can reduce reliability or, in a worst-case scenario, destroy the semiconductor device.

To allow for a rise in junction temperature, manufacturers usually provide derating characteristics, like that shown in Fig.7.5. Note that in this example the device is rated for full power dissipation for temperatures up to 25°C, but thereafter the device is de-rated in a linear fashion up to an absolute maximum junction temperature of 175°C.

Determining temperature rise above ambient

The ability of a heatsink and transistor mounting arrangement to dissipate heat can be measured in terms of the amount by which the junction temperature exceeds the ambient (or surrounding air) temperature. A perfect heatsink and mounting arrangement would keep the junction temperature the same as that of the surrounding air but in a real situation the junction temperature will always rise under operational conditions. What is important, however, is that the maximum junction temperature (T_{Jmax}) must *never* be exceeded.

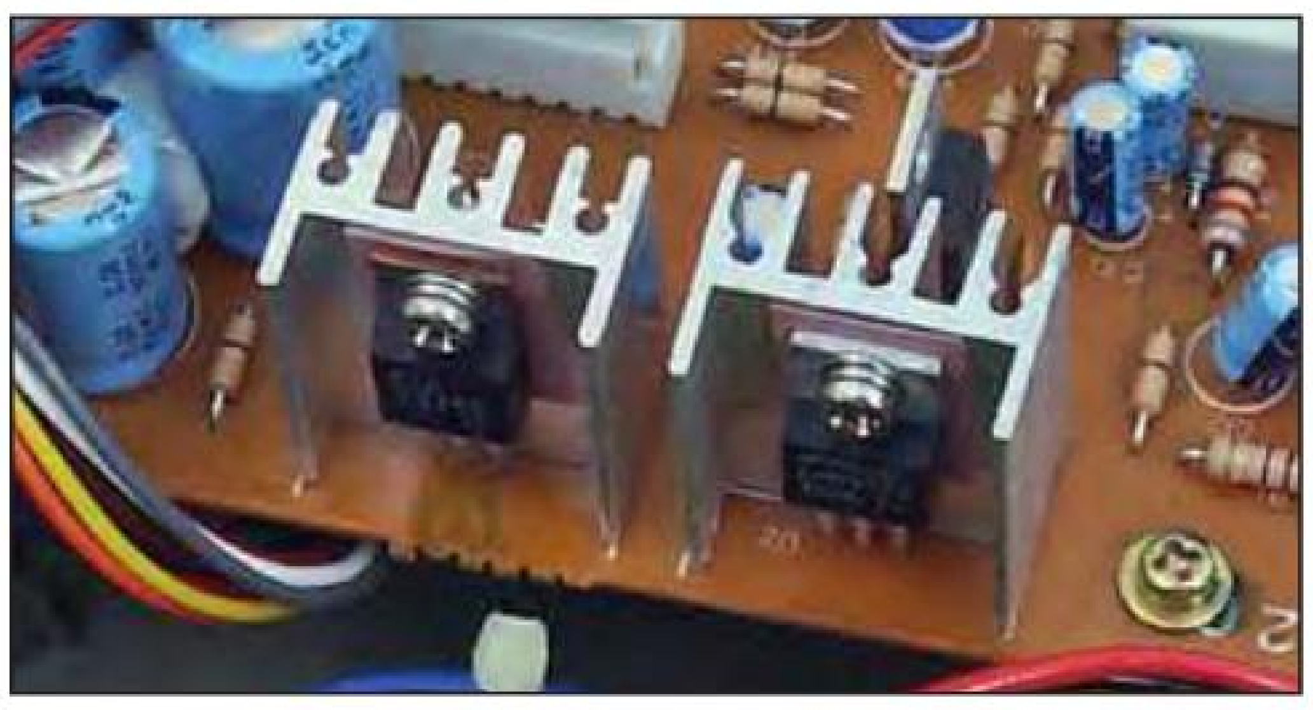


Fig.7.4. Two three-terminal voltage regulators fitted with PCB-mounted heatsinks.

Table 7.1 Algebraic symbols used in heatsink calculations

Symbol	Meaning	Units
T_A	Ambient temperature	°C
T_{Amax}	Maximum ambient temperature	°C
T_J	Junction temperature	°C
T_{Jmax}	Maximum junction temperature	°C
T_C	Case temperature	°C
T_S	Heatsink surface temperature (note 1)	°C
θ_{JA}	Thermal resistance (from junction to ambient)	°C/W
θ_{CS}	Thermal resistance (from case to surface)	°C/W
θ_{SA}	Thermal resistance (from surface to ambient)	°C/W
θ_T	Total thermal resistance (from junction to ambient)	°C/W
P_T	Total power dissipation (note 2)	W
P_{Tmax}	Maximum total power dissipation (note 2)	W

Notes

1. At the point of contact with the semiconductor device
2. For one or more devices mounted on the heatsink

The temperature rise (ΔT) above ambient is given by:

$$\Delta T = P_T \times \theta_T \tag{1}$$

where P_T is the total power dissipated by the semiconductor device(s) and θ_T is the total thermal resistance of the heatsink and mounting arrangement.

Determining actual junction temperature

The temperature rise is simply the difference between the actual junction temperature and the surrounding (or ambient) temperature. Ultimately, the operating junction temperature is vitally important since exceeding the maximum operating junction temperature (T_{Jmax}) will often result in the destruction of the semiconductor device.

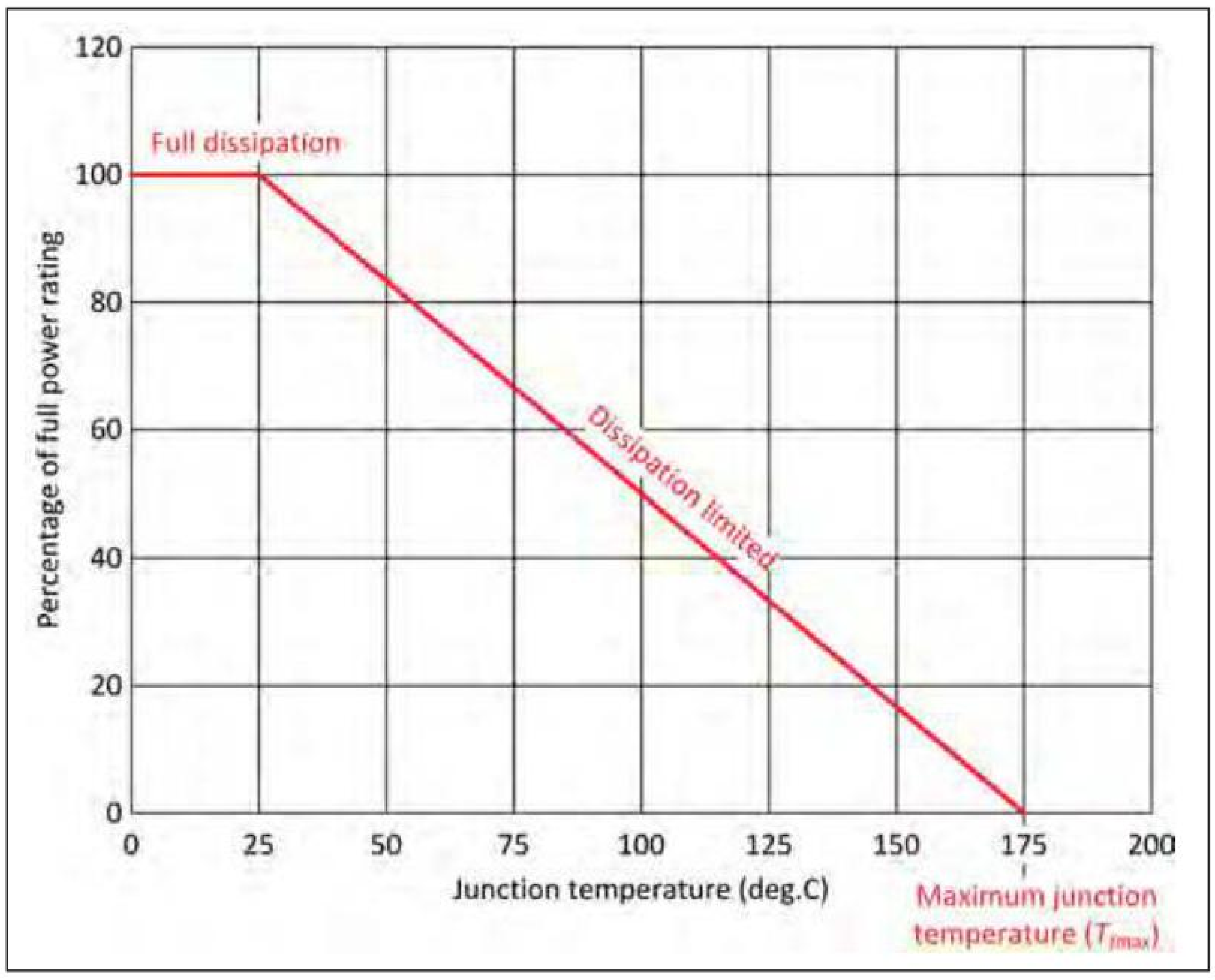


Fig.7.5. Typical derating characteristic.

Example 1

A TO220 transistor dissipates a total power of 5W. If the total thermal resistance is 8°C/W and the ambient temperature is 30°C, determine the junction temperature.

Applying equation (3) gives:

$$T_J = (P_T \times \theta_T) + T_A = (5 \times 8) + 30 = 70°C$$

$$\text{Now } \Delta T = T_J - T_A \qquad \textbf{(2)}$$

Combining equations (1) and (2) gives:

$$T_J - T_A = P_T \times \theta_T$$

From which:

$$T_J = (P_T \times \theta_T) + T_A \qquad \textbf{(3)}$$

It is worth using an example to explain the use of this formula – see example 1, above.

Determining thermal resistance

Fig.7.6 shows a typical TO3-case-style transistor mounted on a finned heat dissipator. Heat is conducted away from the semiconductor junction to the outer case of the TO3 package and then, via an insulating washer to the surface of the heatsink. From this point, heat is conducted to the extremities of the fins where it is radiated into the surrounding air space.

The total thermal resistance (θ_T) present in any heatsink arrangement is the sum of several individual thermal resistances. Consider the case of a transistor bolted directly to a metal heat radiator (see Fig.7.7). The total thermal resistance (θ_T) is the sum of the following thermal resistances:

- The thermal resistance that exists between the semiconductor junction and the case of the transistor (ie, the thermal resistance *inside* the transistor package), θ_{JC}
- The thermal resistance between the case of the transistor and the heat radiator (ie, the thermal resistance of an insulating washer, where it is fitted), θ_{CS}
- The thermal resistance between the surface of the heat radiator and the space surrounding it (ie, the thermal resistance between the surface and ambient), θ_{SA}

Fig.7.8 shows these three thermal resistances together with the temperatures that exist at each point in the arrangement shown in Fig.7.6.

The three thermal resistances shown in Fig.7.8 appear in series and we can use a simple electrical analogy to represent the thermal equivalent circuit in electrical terms, as shown in Fig.7.9.

Example 2

A transistor has a thermal resistance from junction to case of 1.5°C/W. If the transistor is fitted with a washer and mounting kit having a thermal resistance of 1.75°C/W, and a heatsink of 2.25°C/W, determine the total thermal resistance that exists between junction and ambient.

Applying equation (5) gives:

$$\theta_{JA} = \theta_{JC} + \theta_{CS} + \theta_{SA} = 1.5 + 1.75 + 2.25 = 5.5°C/W$$

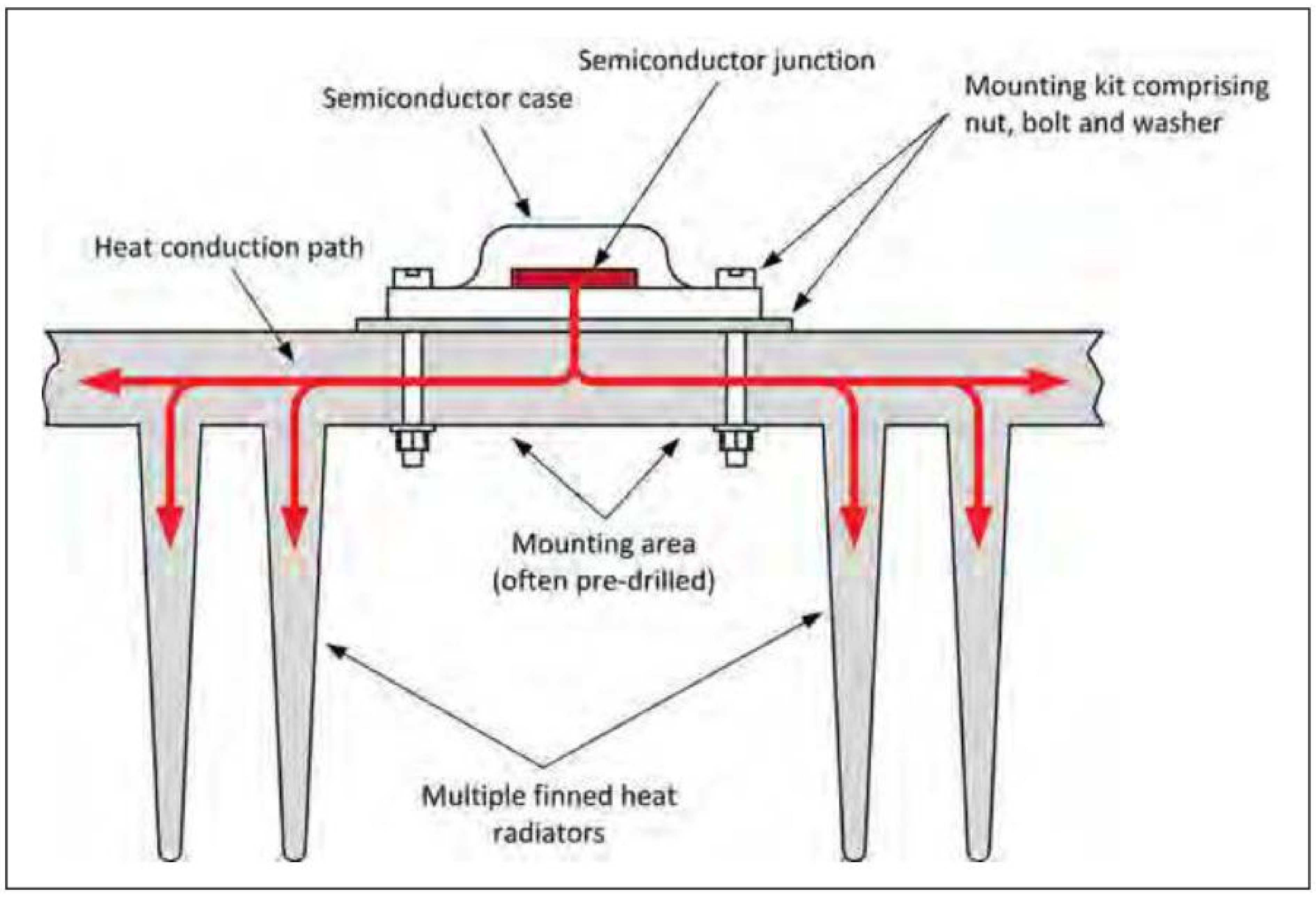

Semiconductor junction
Semiconductor case
Mounting kit comprising nut, bolt and washer
Heat conduction path
Mounting area (often pre-drilled)
Multiple finned heat radiators

Fig.7.6. Typical transistor mounting arrangement showing paths for conducted heat.

Note that the source of power (P_{TOT}) is the semiconductor device and the 'potentials' at the two extreme ends of the series chain of thermal resistances, which are T_J (junction temperature) and T_A (ambient or surrounding air temperature).

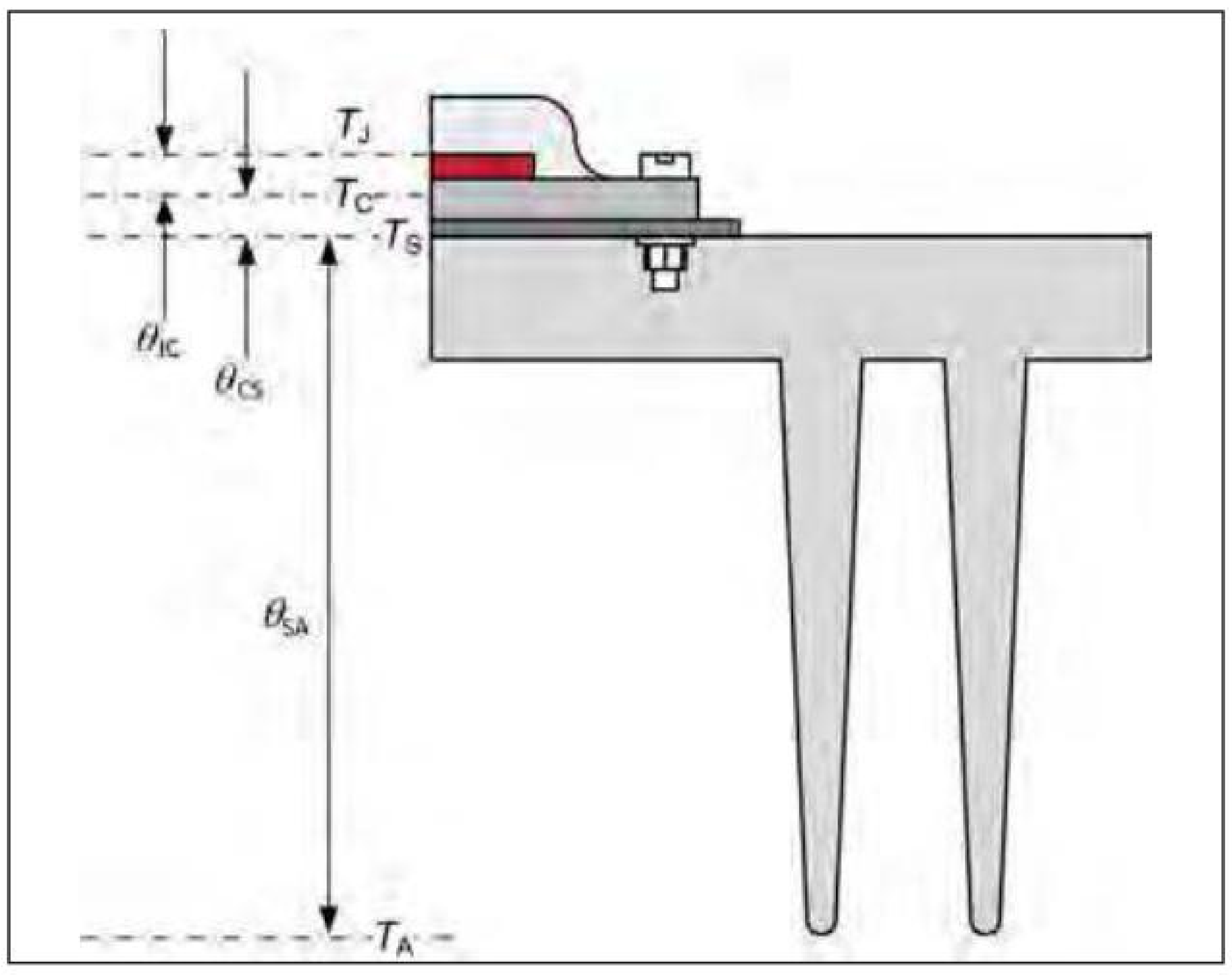


Fig.7.7. Temperatures (T_J, T_C and T_S) and thermal resistances (θ_{JC}, θ_{CS} and θ_{SA}) present in Fig.7.6.

From Fig.7.9 we can conclude that the total thermal resistance (θ_T) is given by:

$$\theta_T = \theta_{JC} + \theta_{CS} + \theta_{SA} \qquad \textbf{(4)}$$

Note that the total thermal resistance is the same as the thermal resistance between the junction and ambient, thus:

$$\theta_T = \theta_{JC} + \theta_{CS} + \theta_{SA} = \theta_{JA} \qquad \textbf{(5)}$$

Typical thermal resistances

The typical thermal resistances of some common semiconductor case styles are listed in Table 7.2.

Table 7.2 Typical thermal resistances for different semiconductor case styles

Case style	θ_{JA} (°C/W) (no heatsink)	θ_{JC} (°C/W)
TO92	200 to 350	40 to 60
TO126	83 to 100	3 to 10
TO220	60 to 70	1.5 to 4
TO202	62 to 75	6 to 13
TO218	30 to 45	1 to 1.56

Having got some of the essential mathematics out of the way let's consider another calculation – see example 2.

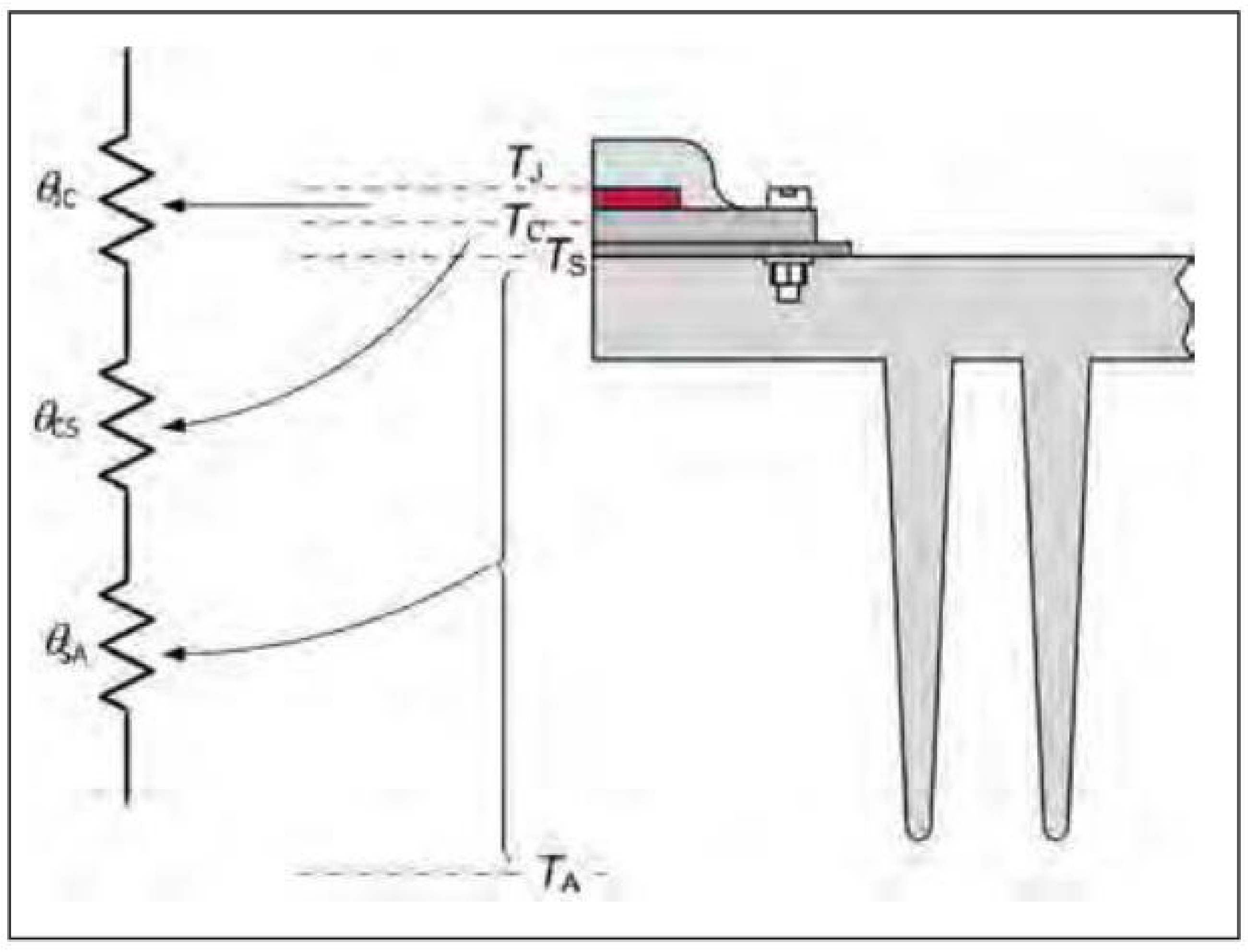


Fig.7.8. Thermal resistances present in Fig.7.7.

Worst-case conditions

In the design of electronic equipment we usually need to plan for the worst-case conditions, ensuring that the absolute maximum junction temperature (T_{Jmax}) is not exceeded when

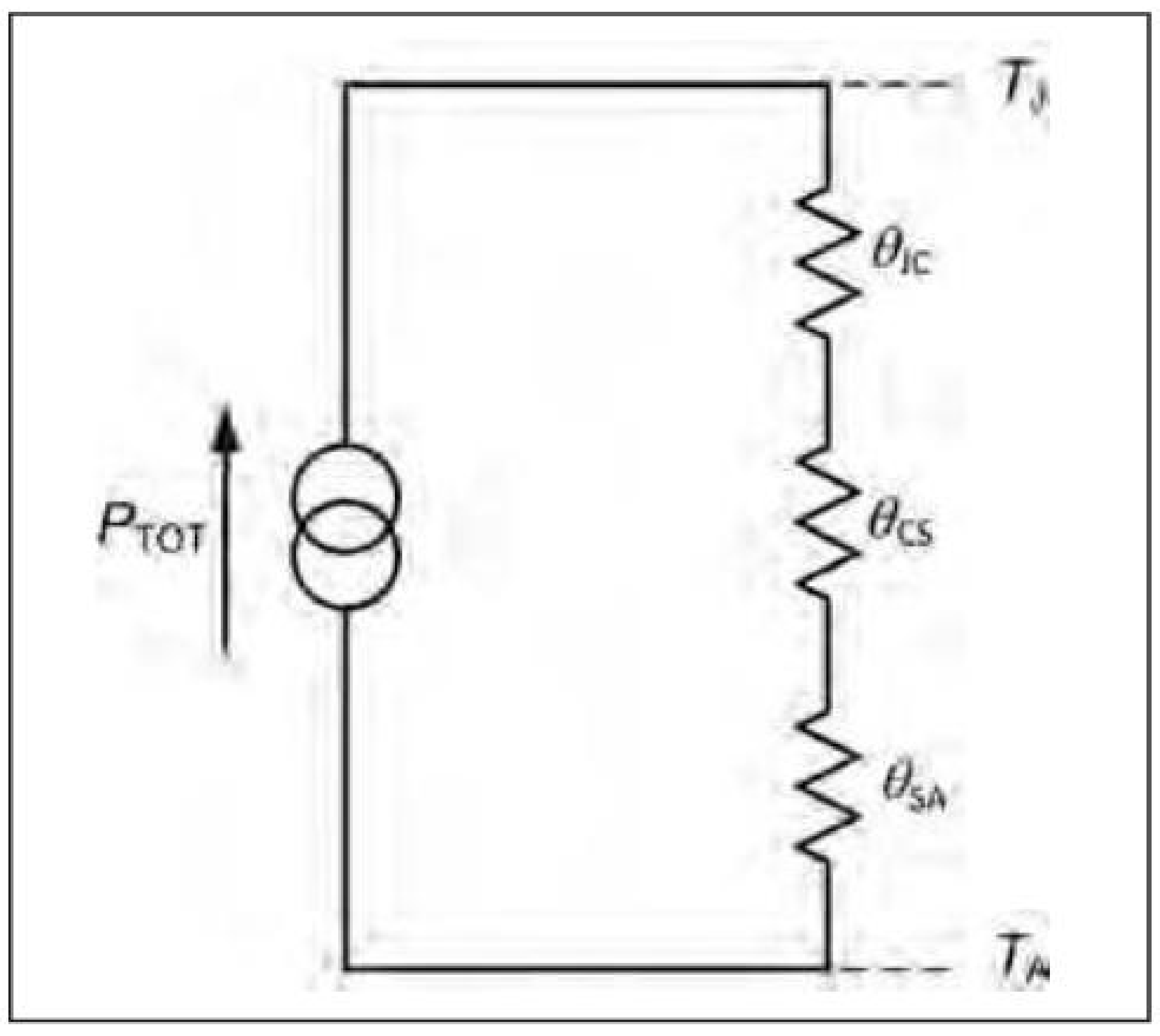


Fig.7.9. Thermal equivalent circuit.

the total power dissipation and ambient temperature reach their maximum working values.

From equation (3) we can infer that, under the worst-case conditions:

$$T_{J\max} = (P_{T\max} \times \theta_T) + T_{A\max}$$

Combining this with equation (5) gives:

$$T_{J\max} = (P_{T\max} \times (\theta_{JC} + \theta_{CS} + \theta_{SA})) + T_{A\max} \qquad \textbf{(6)}$$

We will illustrate this with example 3.

Determining heatsink specifications

The electronic equipment designer often has to determine the required heatsink specifications given parameters such as the absolute maximum junction temperature, thermal resistance from junction to case and maximum expected ambient temperature. To do this, we need to rearrange equation (6) to make θ_{SA} the subject of the equation. Thus:

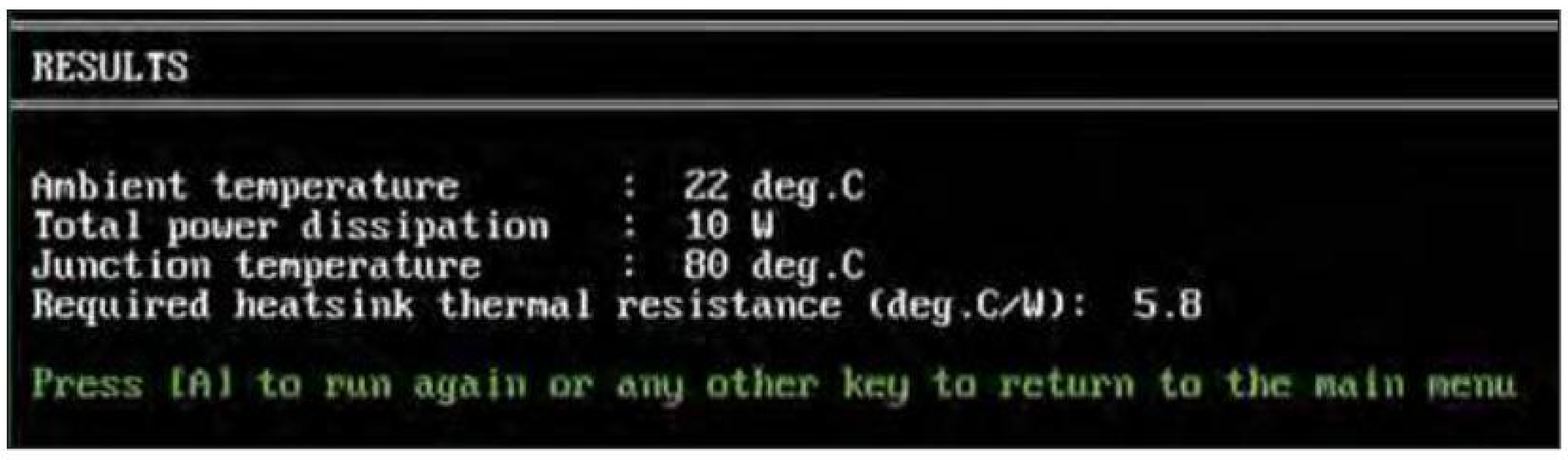
$$\theta_{SA} = \frac{T_{J\max} - T_{A\max}}{P_{T\max}} - (\theta_{JC} + \theta_{CS}) \qquad \textbf{(7)}$$

The value obtained for θ_{SA} will be the minimum acceptable rating for the required heatsink. In practice, we would choose a component with a higher rating to allow for a margin of safety. Examples 4 and 5 show how this works.

Heatsink design software

While these calculations are not particularly difficult, they can become a little tedious if you need to optimise several devices. Fig.7.10 shows a simple BASIC program (available for download from the *PE* website) that can be used to assist with heatsink calculations. This Windows-based software provides three main options:

- Calculation of required heatsink thermal resistance given various operating conditions
- Calculation of operating junction temperature given heatsink and other parameters
- Setting default values for maximum junction temperature and thermal resistance.



```
RESULTS

Ambient temperature          :  22 deg.C
Total power dissipation      :  10 W
Junction temperature         :  80 deg.C
Required heatsink thermal resistance (deg.C/W):  5.8

Press [A] to run again or any other key to return to the main menu
```

Fig.7.10. Using software to assist with heatsink calculations.

Example 3

A transistor has an absolute maximum junction temperature rating of 150° and a thermal resistance from junction to case of 1.0°C/W. If the device is fitted with a washer and mounting kit having a thermal resistance of 1.25°C/W and a heatsink of 2.75°C/W, determine whether the maximum ratings are exceeded when the total power dissipation reaches a maximum of 25W at a maximum ambient temperature of 40°C.

Applying equation (6) gives:

$$T_{J\max} = (25 \times (1.0 + 1.25 + 2.75)) + 40 = 165°\text{C}$$

The absolute maximum junction temperature rating (150°C) is exceeded and the designer should either reduce the power dissipation to a safe value or reduce the thermal resistance (or both).

Example 4

Determine the minimum acceptable thermal resistance rating for a heatsink for use with a transistor under the following conditions:

Maximum junction temperature	175°
Thermal resistance of case to surface	1.25°C/W
Thermal resistance washer/mounting kit	1.75°C/W
Maximum total power dissipation	13W
Maximum ambient temperature	45°C

Applying equation (7) gives:

$$\theta_{SA} = \frac{T_{J\max} - T_{A\max}}{P_{T\max}} - (\theta_{JC} + \theta_{CS}) = \frac{175 - 45}{13} - (1.25 + 1.75)$$

$$= 10 - 3 = 7°\text{C/W}$$

Example 5

A power FET operates with a maximum continuous drain current (I_D) of 0.5A at a maximum continuous drain-source voltage (V_{DS}) of 40V. Find the minimum acceptable thermal resistance rating for a heatsink for use with the device under the following conditions:

Maximum junction temperature	175°
Thermal resistance of case to surface	1.35°C/W
Thermal resistance washer/mounting kit	1.45°C/W
Maximum ambient temperature	55°C

First, we need to determine the maximum total power dissipation. Since the device is a FET, the gate dissipation can be neglected. Thus, the total power dissipation will be:

$$P_{T\max} = V_{DS} \times I_D = 40 \times 0.5 = 20\text{W}$$

Next, applying equation (7) gives:

$$\theta_{SA} = \frac{T_{J\max} - T_{A\max}}{P_{T\max}} - (\theta_{JC} + \theta_{CS}) = \frac{175 - 55}{20} - (1.35 + 1.45)$$

$$= 6 - 2.8 = 3.2°\text{C/W}$$

Low-current Negative Voltage Supply Converter

This month's *Practical Project* is a *Low-current Negative Supply Converter* based on a low-cost 555 timer. A typical application for the circuit is supplying a nominal –5V at up to 30mA from a 12V DC supply. The specifications for this *Practical Project* are in the following table:

Input voltage	12V (nominal)
Output voltage	–5V ±0.5V (with Zener regulator)
Load current	15mA nominal (30mA max.)
Switching frequency	32kHz approx.
Load regulation	13% approx.
Output resistance	40Ω approx.

The *Low-current Negative Supply Converter* circuit is shown in Fig.7.11. A 555 timer (IC1) is used as an astable oscillator with its operating frequency determined by R1, R2 and C2. The square wave output from pin-3 at approximately 32kHz (see Fig.7.14) is fed to a simple rectifier-doubler arrangement comprising D2/D3 and C4/C5. The final DC output appearing across C5 will be just less than the supply voltage but of opposite polarity.

You will need...

Perforated copper stripboard (9 strips each with 25 holes)
2 2-way PCB screw terminal connector (ST1 and ST2)
1 1kΩ resistor (R1)
1 22kΩ resistor (R2)
3 100µF 35V capacitors (C1, C4 and C5)
1 1nF ceramic capacitor (C2)
1 100nF ceramic capacitor (C3)
1 555 8-pin DIL timer (IC1)
1 low-profile 8-pin DIL socket
3 1N4148 diodes (D1 to D3)
1 5.1V Zener diode (optional, see text)
4 stand-off pillars and mounting screws

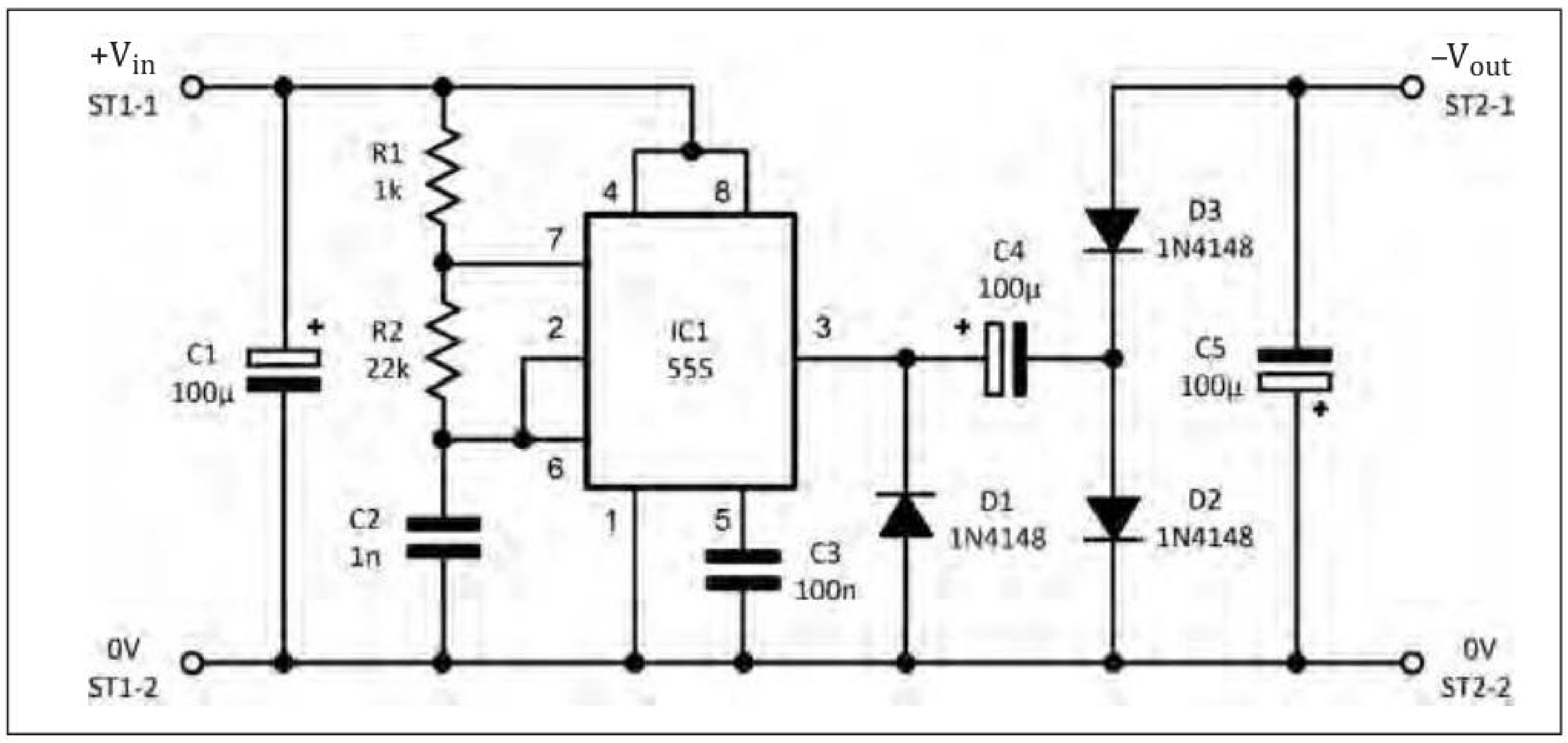

+V$_{in}$
ST1-1
–V$_{out}$
ST2-1
R1
1k
4
8
7
D3
1N4148
C4
100µ
C1
100µ
R2
22k
2
IC1
555
3
C5
100µ
6
1
5
D1
1N4148
D2
1N4148
C2
1n
C3
100n
0V
ST1-2
0V
ST2-2

Fig.7.11. Teach-In 2019 Practical Project: Low-current Negative Supply Converter.

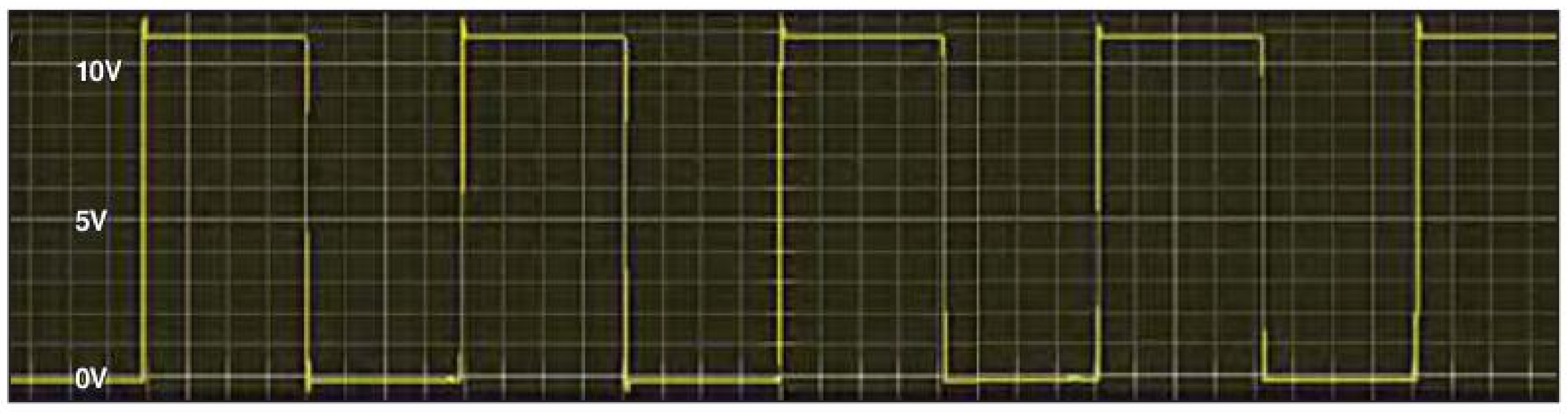


Fig.7.14. Switching waveform at pin-3 of IC1 (note: this is an almost symmetrical square wave).

Construction

The stripboard layout of the *Low-current Negative Supply Converter* is shown in Fig.7.12. Note that there are 24 track breaks on the underside of the board and 13 links on the upper side. As always, once assembly is complete it is well worth carrying out a careful inspection of the circuit board, particularly checking the off-board wiring and links to ST1 and ST2.

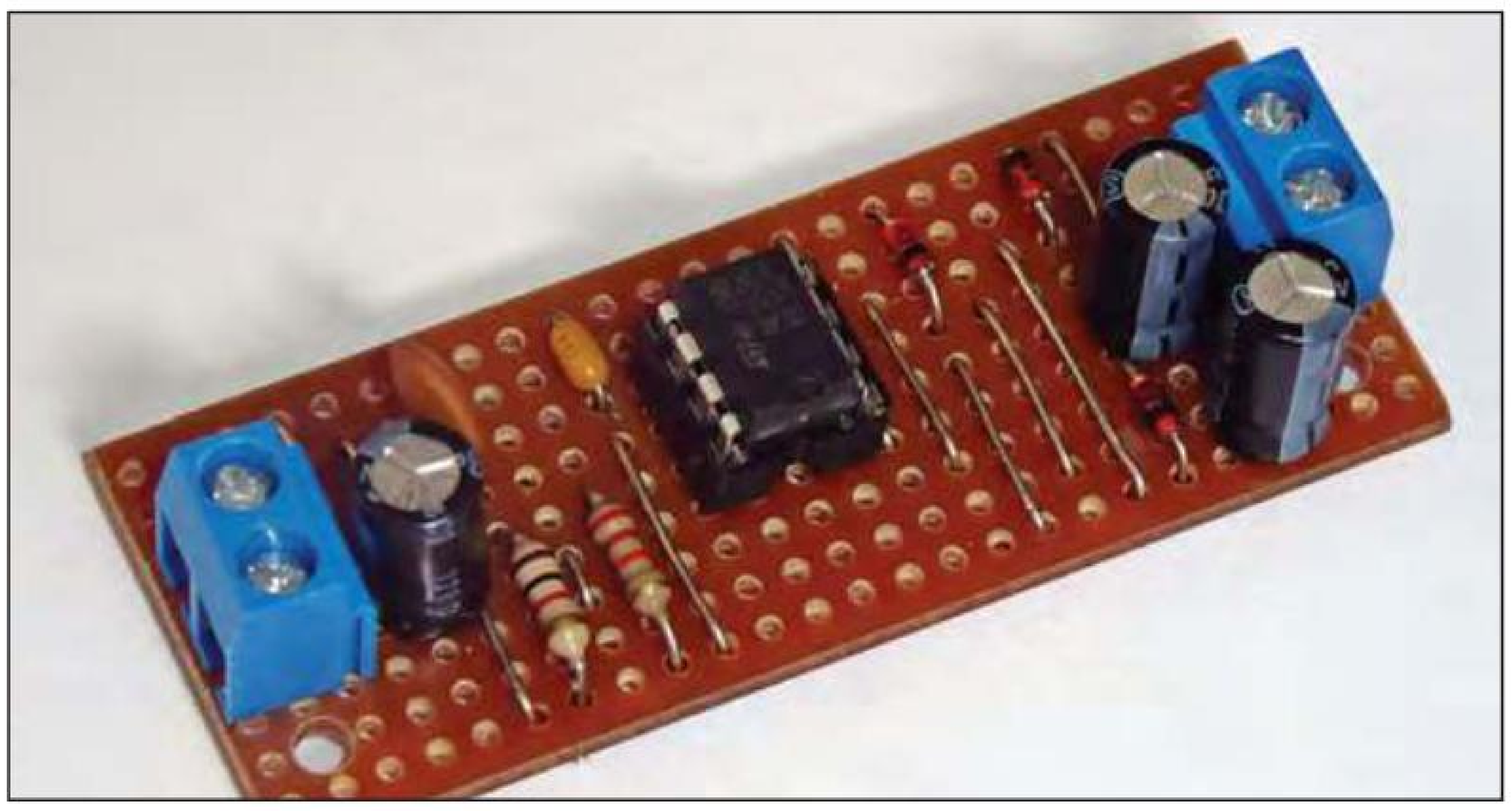


Fig.7.13. The finished Low-current Negative Supply Converter.

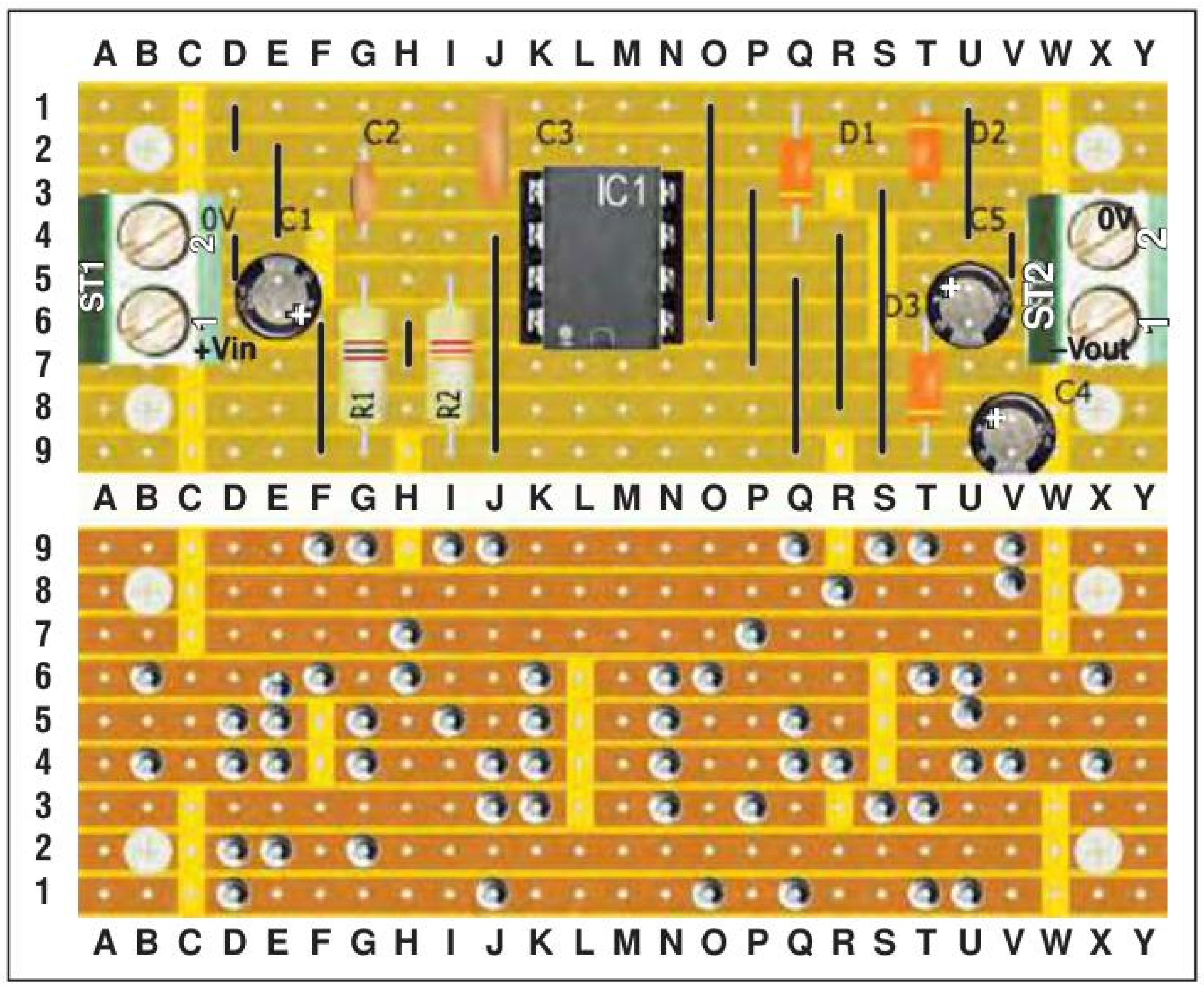

A B C D E F G H I J K L M N O P Q R S T U V W X Y
1
2
3
4
5
6
7
8
9
C2
C3
IC1
D1
D2
0V
C1
ST1
2
1
+Vin
R1
R2
D3
C5
0V
ST2
2
1
–Vout
C4
A B C D E F G H I J K L M N O P Q R S T U V W X Y
9
8
7
6
5
4
3
2
1
A B C D E F G H I J K L M N O P Q R S T U V W X Y

Fig.7.12. Stripboard layout of the Low-current Negative Supply Converter.

The off-load output voltage of the *Low-current Negative Supply Converter* will fall in magnitude as the output is loaded (see Fig.7.16). This may not be a problem in some applications but in others it may be essential to maintain the output much closer to the nominal –5V. In such cases, a 5.1V shunt Zener diode can simply be connected in parallel with C5, as shown in Fig.7.15. The off-load output voltage will then be near to –5.1V, falling to –4.5V as the nominal 15mA load current is reached (see Fig.7.16).

Next month

In next month's *Teach-In 2019* we will bring this series to a close with a look at battery types and characteristics. Our *Practical Project* will take the form of a handy battery monitor. We also include a full index to the series.

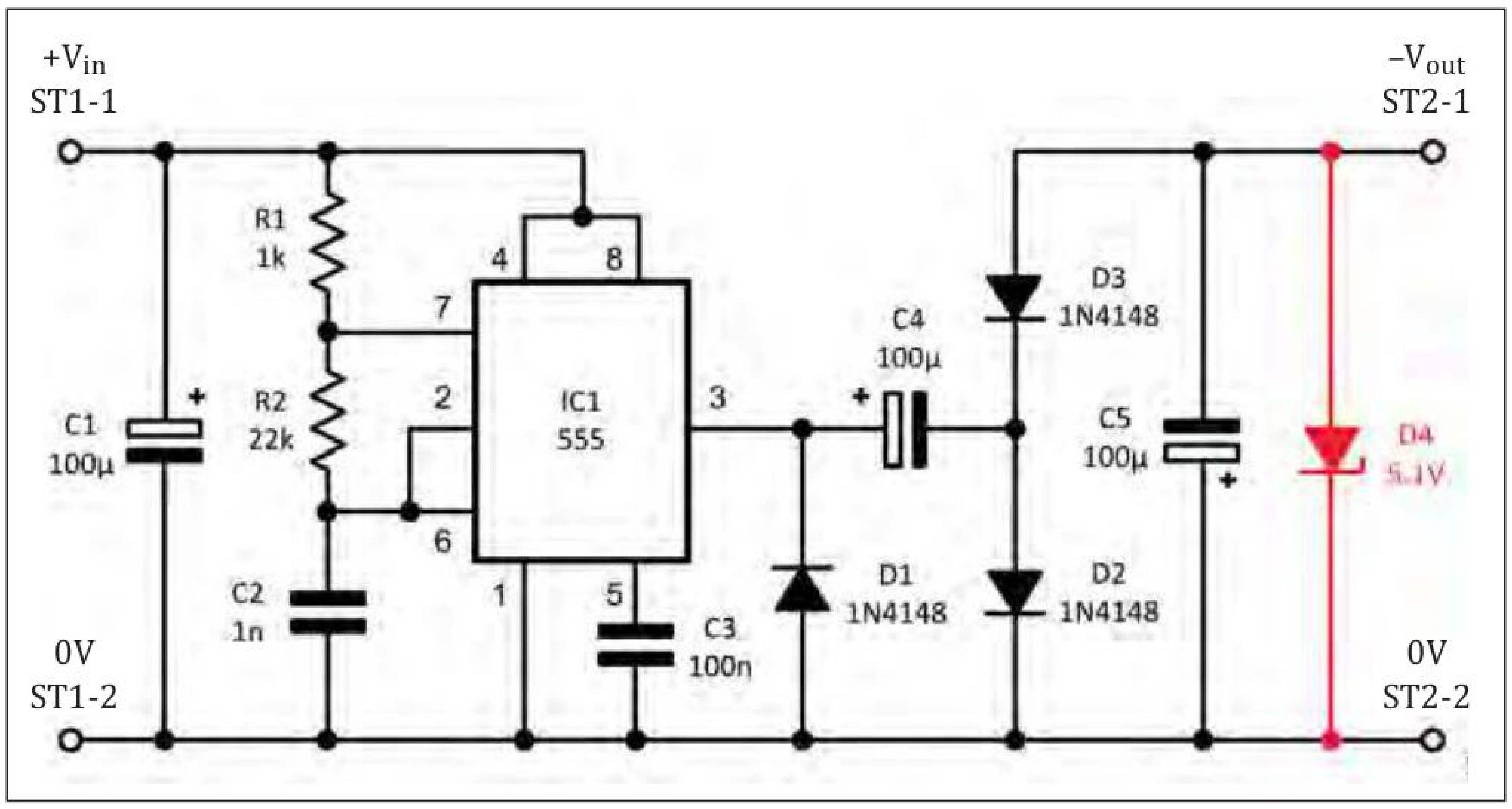

+V$_{in}$
ST1-1
–V$_{out}$
ST2-1
R1
1k
4
8
7
D3
1N4148
C4
100µ
C1
100µ
R2
22k
2
IC1
555
3
C5
100µ
D4
5.1V
6
1
5
D1
1N4148
D2
1N4148
C2
1n
C3
100n
0V
ST1-2
0V
ST2-2

Fig.7.15. 5.1V Zener regulator added to the Low-current Negative Supply Converter.

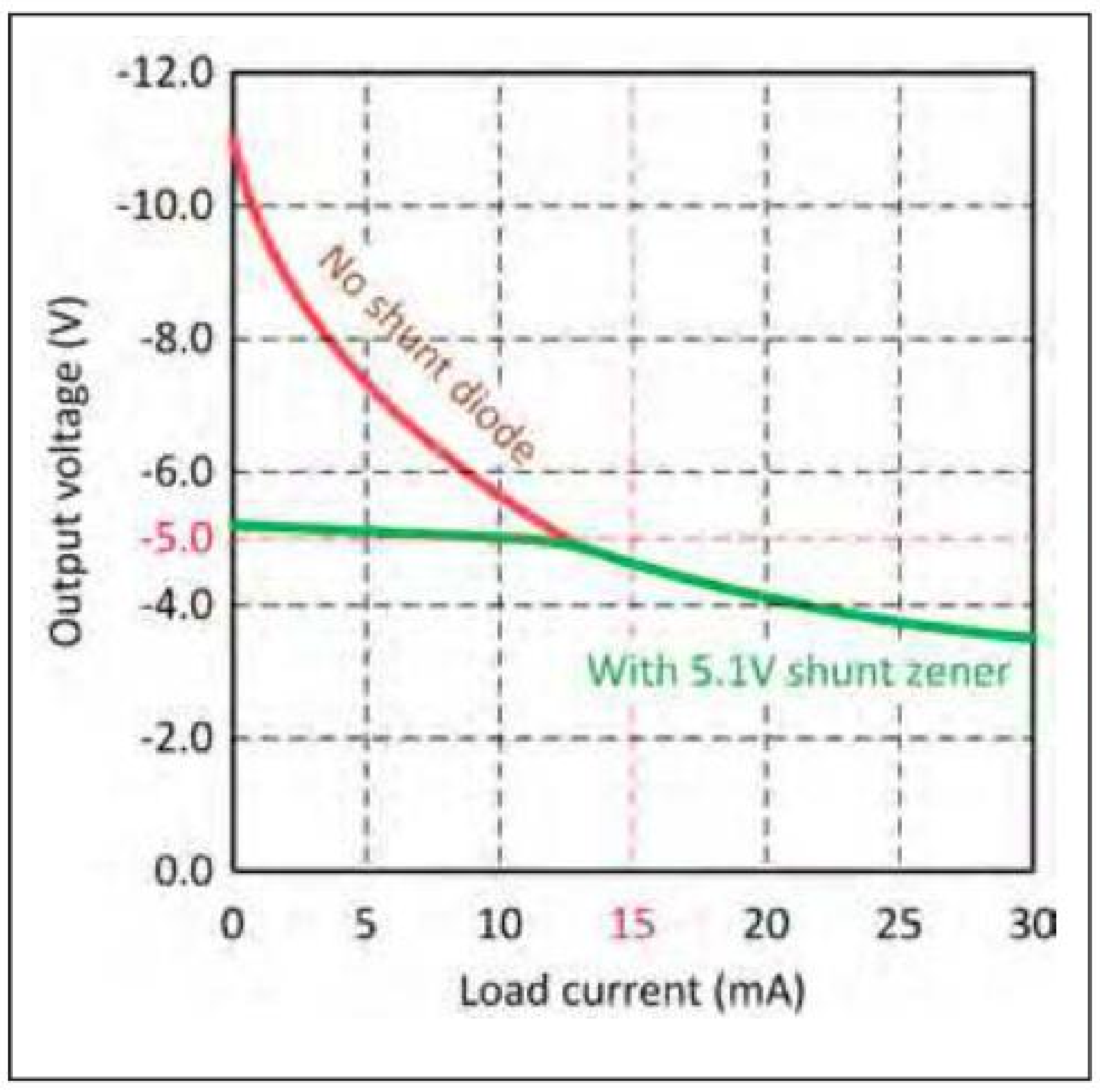
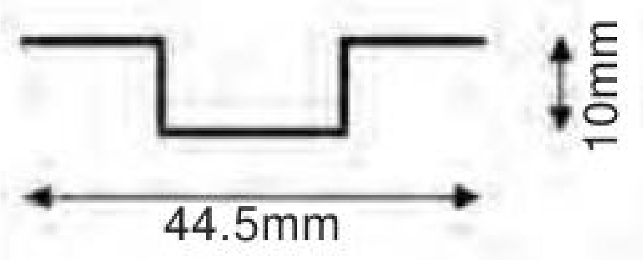
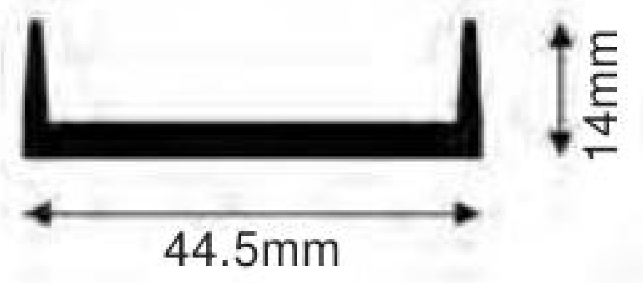


Fig.7.16. Load regulation characteristic for the Low-current Negative Supply Converter.

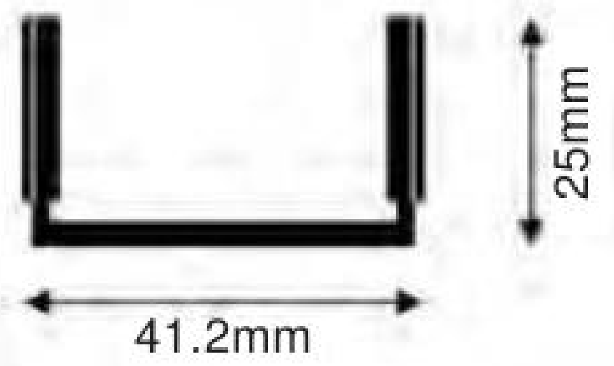
Get it Right! – Selecting and using heatsinks



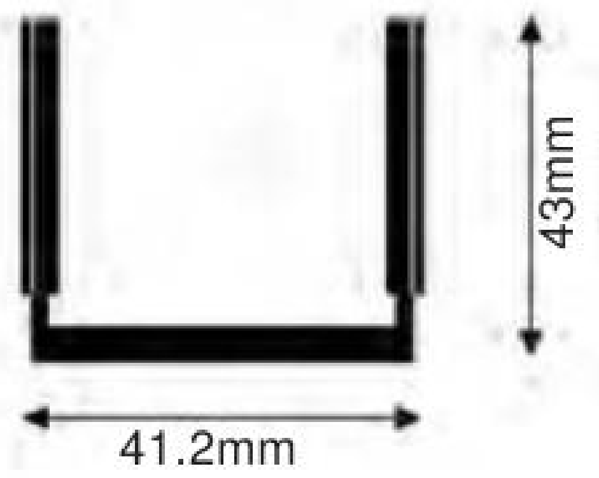
Length (mm)	Thermal resistance (°C/W)
17.8	20.0



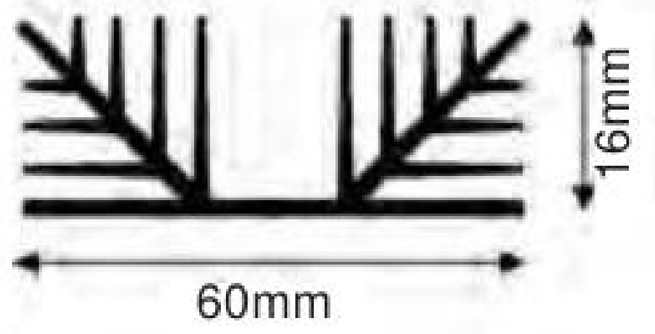
Length (mm)	Thermal resistance (°C/W)
32	14.0



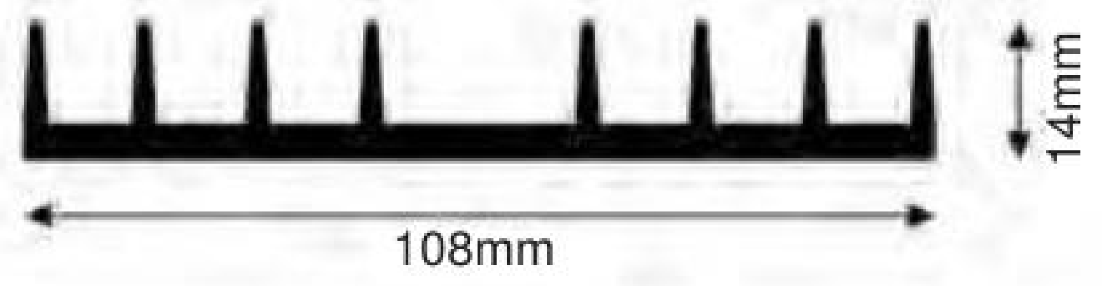
Length (mm)	Thermal resistance (°C/W)
38	7.1



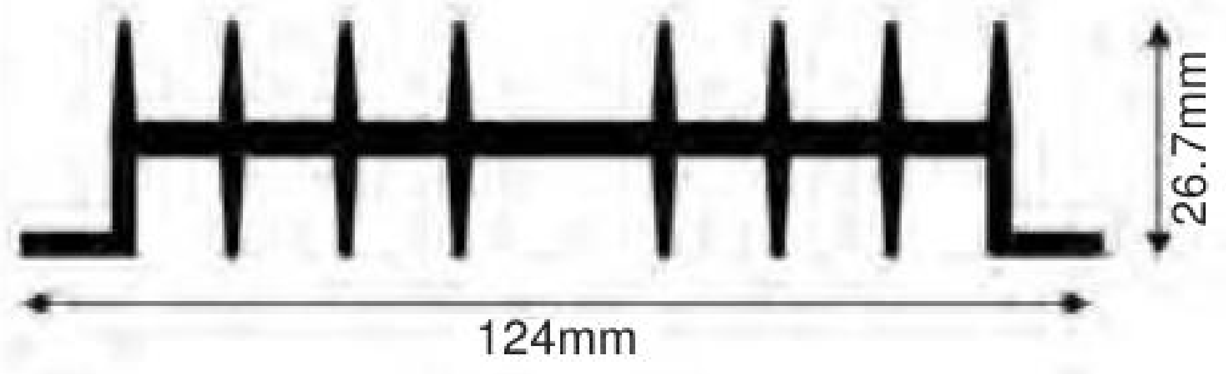
Length (mm)	Thermal resistance (°C/W)
45	4.0



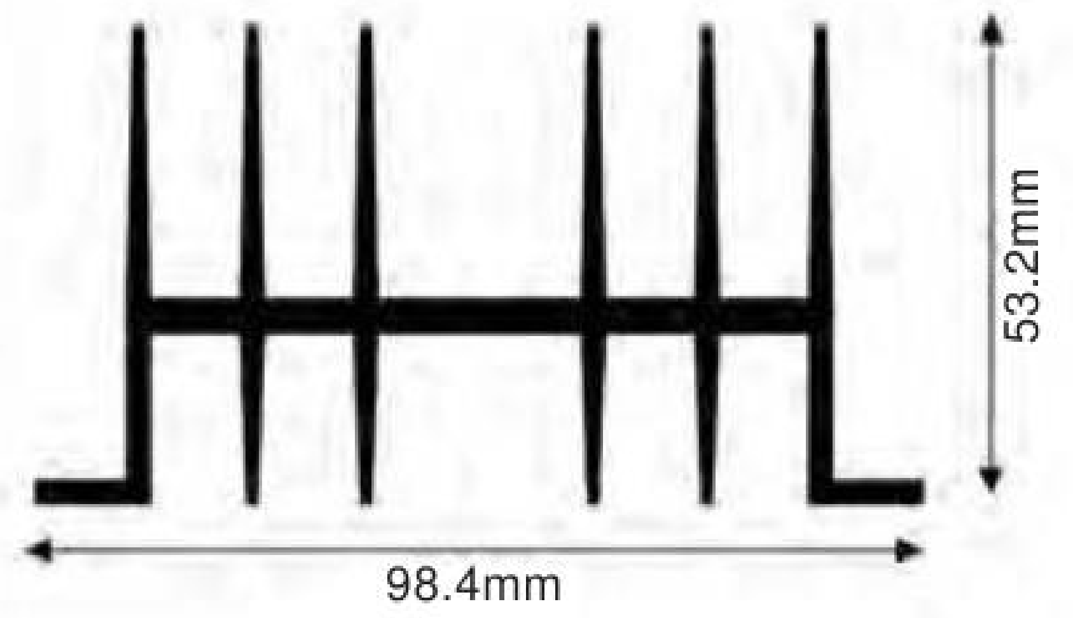
Length (mm)	Thermal resistance (°C/W)
51	6.2
89	1.40



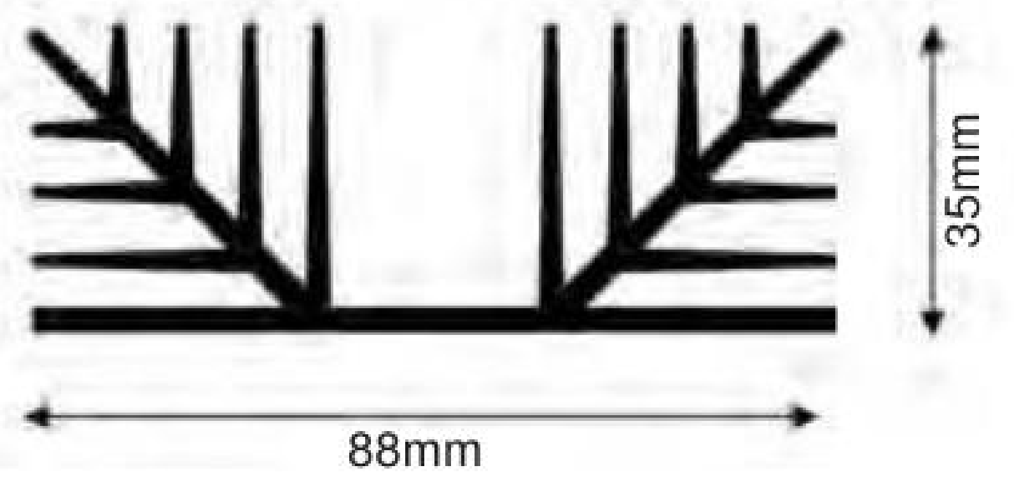
Length (mm)	Thermal resistance (°C/W)
50	5.0
75	3.9
87.5	3.5
100	3.2
150	2.6
300	1.75



Length (mm)	Thermal resistance (°C/W)
75	2.72
100	2.3
150	1.85



Length (mm)	Thermal resistance (°C/W)
50	50
75	2.32
100	1.97
150	1.58



Length (mm)	Thermal resistance (°C/W)
75	1.65
100	1.40
125	1.20

Fig.7.17. Heatsink profiles, dimensions and typical thermal resistances.

Having dealt with the theory of heatsink design it is worth mentioning the following important practical considerations.

Estimating thermal resistance

It is sometimes necessary to estimate the thermal resistance of a heatsink from its construction and dimensions. To assist with this task, Fig.7.17 shows a selection of heatsink profiles along with dimensions and typical thermal resistances. To use the chart, simply identify the type, approximate dimensions and cross-sectional profile of your heatsink and then refer to the thermal resistance tables.

Mounting arrangements

A typical mounting arrangement for a TO220 semiconductor package is shown in Fig.7.18. In many cases, the tab of the device is connected to one of three terminals (often the collector or drain), in which case an electrically insulating but thermally conductive mica or plastic washer must be fitted. Note also that an insulated bush must be used in order to prevent the mounting bolt shorting the metal tab to the heatsink (see Fig.7.18).

The thermal resistance of the mounting kit used with a semiconductor device can have a major effect on the efficiency of the heat conduction from the surface of the case to the heat radiator. Special thermally impregnated washers have significantly lower thermal resistance than simple mica washers. Thermally conductive silicone grease should not be used with this type of washer.

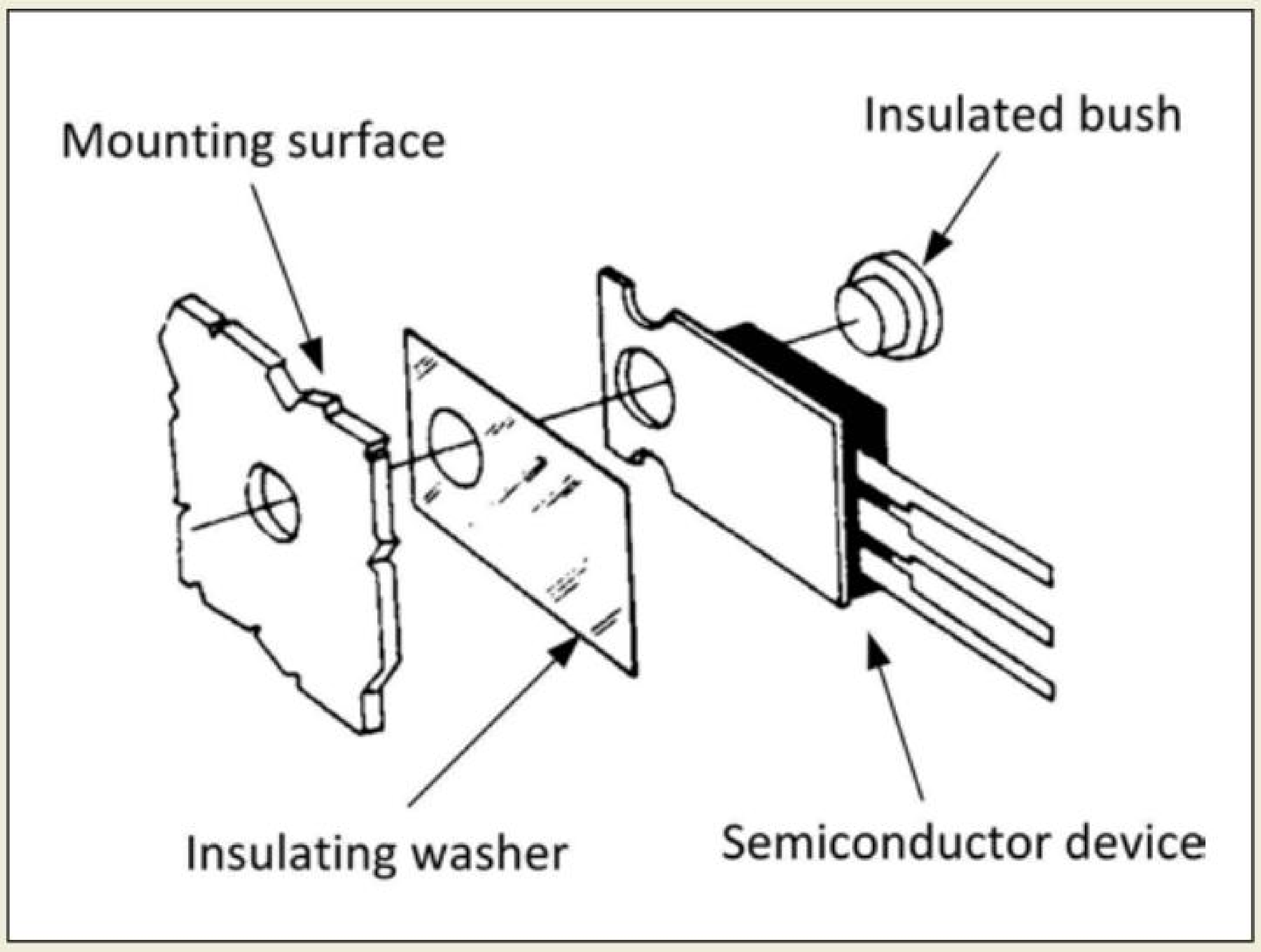


Fig.7.18. Use of insulating washers and bushes.

Airflow

It is important to note that calculations assume that a heatsink is subject to natural convection. Increasing the airflow across the surface of the heatsink can dramatically increase the amount of dissipated heat and consequently it will also significantly reduce the effective thermal resistance of the heatsink. Thus, you might also need to:

1. Ensure adequate airflow around a heatsink, drawing cool air into an enclosure and allowing warm air to flow out again.
2. Consider the use of forced-air / fan-assisted cooling in cases where there are space restrictions or where ventilation is problematic.
3. Finally, natural convection airflow can be enhanced by the proper placement of heatsinks and other heat producing components. Warm air rises, so vertical surfaces transmit heat to the air better than comparable horizontal surfaces. The hottest devices should be located on the upper side of a horizontally mounted PCB or close to the upper edge of a vertically mounted PCB.

AUDIO OUT

By Jake Rothman

The PE Theremin – Part 1

The *PE Theremin*, which we will build in upcoming issues. (Photo: Harvey Rothman)

There have been plenty of Theremin projects published in *PE*/*EPE* over the years, and with the Theremin's centenary coming up this year, we'll celebrate with another version of this enigmatic piece of electronics. I've been building Theremins since 1993, starting with the Banabi 1974 design, and ever since then I've been developing a body of knowledge about their design and construction. This three-part project will provide circuits and construction tips, culminating in a new low-cost design. Those wanting a 'stuff board and play' design will have to wait until I combine all the boards on a single PCB.

The history of the Theremin has been well covered, but has been subject to a little hyperbole. I think that the musical instrument idea came from the hand-capacity whistling effects when tuning early regenerative AM radios. Leon Theremin designed some unique circuitry to exploit hand capacity to make the first useable electronic musical instrument around 1920, shown in Fig.1. His original circuitry, which paved the way for the Theremin, was designed to measure the dielectric constants of gases in a physics lab. Peter Lankshear in *Radio Bygones*, April/May 1994, covered the original RCA production circuit.

The basic system

All 'proper' Theremins (in the historical sense) work by detecting very small changes in capacitance resulting in small changes in a radio frequency oscillator. This is then heterodyned or used to 'beat' with another oscillator to produce a difference frequency that is the musical note, as shown in Fig.2. The oscillators frequencies are typically 140kHz to 2MHz and the beat frequency occurs in the audio musical range of about 30Hz to 4kHz. A metal rod is connected to one oscillator to act as a hand-controlled variable capacitor plate. While technically incorrect, this is often referred to as an 'aerial' or 'antenna'. Incidentally, this heterodyning circuitry is often used in metal detectors, albeit with the detection property being mutual inductance rather than capacitance.

Fig.1. Bruce Woolley, co-writer of hit single *Video Killed the Radio Star*, playing his original 1928 RCA Theremin.

Instability and earthing

The high frequencies 'amplify' the slightest frequency difference between the oscillators, giving an incredibly sensitive response. Unfortunately, this also amplifies any instabilities, making Theremin circuits very prone to detuning and temperature drift. Just moving a Theremin to a different position can result in a tuning tweak being necessary. The earthing arrangements have an important role, forming part of a capacitive coupling circuit between the user and the plates, as shown in Fig.3. ('Ideally', a player has sweaty feet

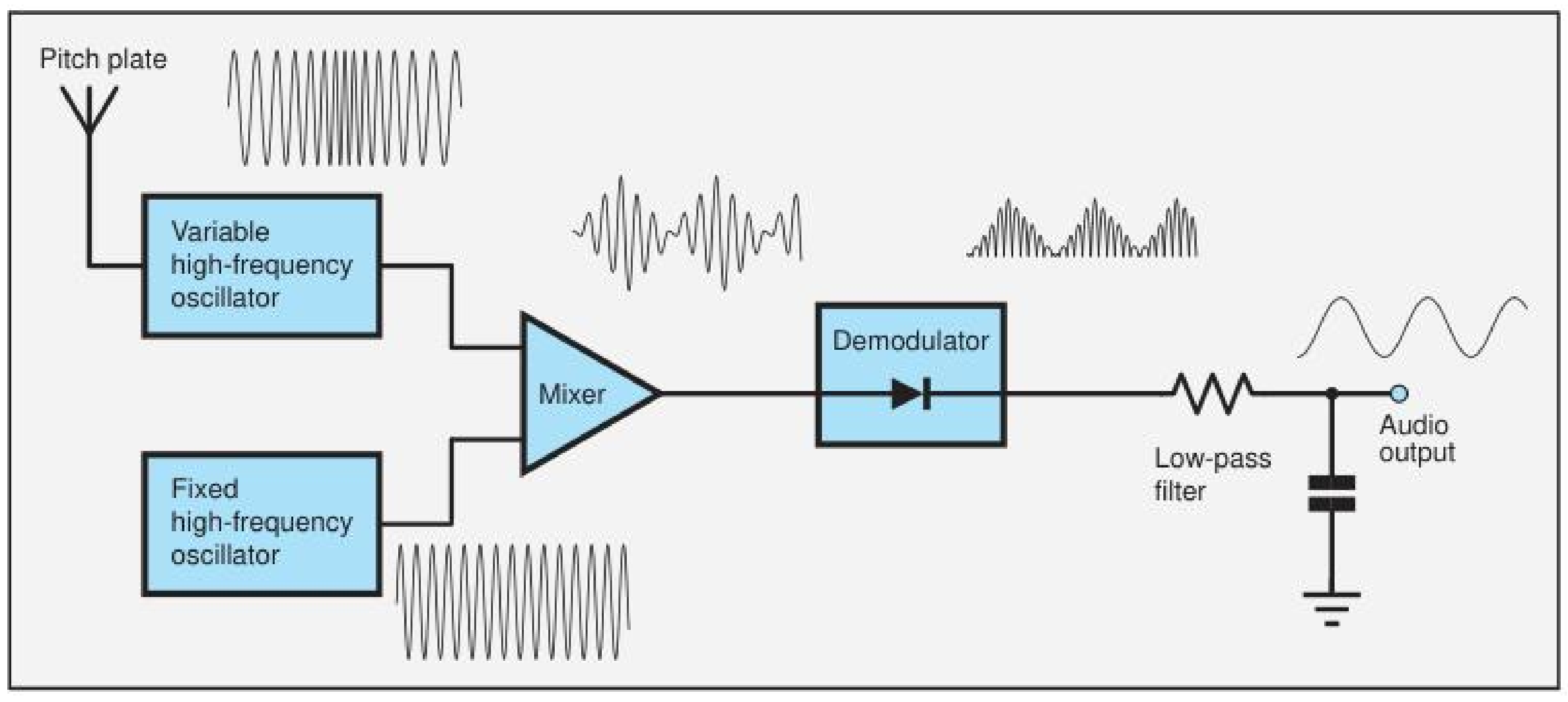

Pitch plate
Variable high-frequency oscillator
Fixed high-frequency oscillator
Mixer
Demodulator
Low-pass filter
Audio output

Fig.2. Block diagram showing the basic Theremin set up.

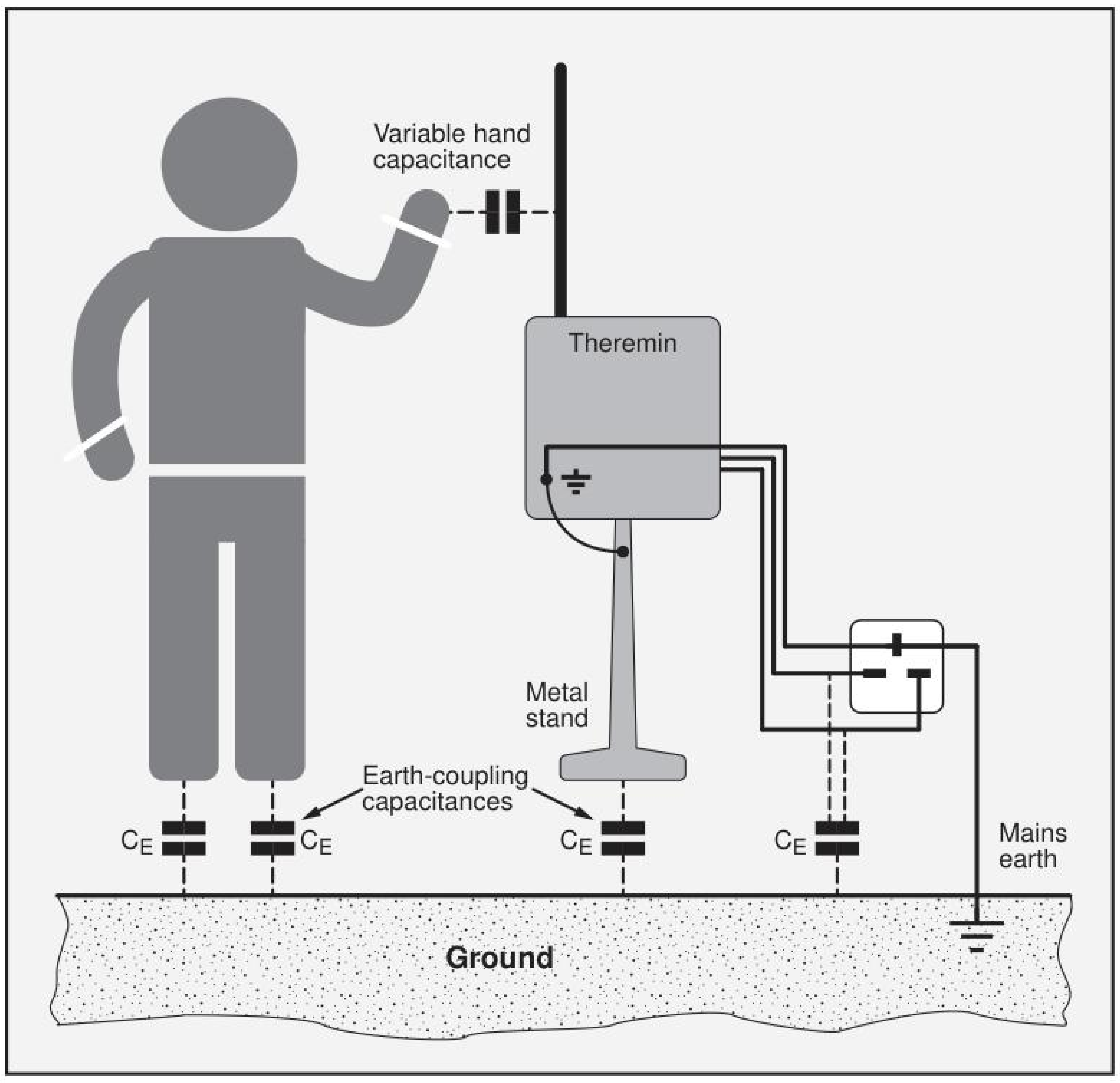


Fig.3. Capacitive circuit involving variable hand capacitance and player/Theremin coupling to ground.

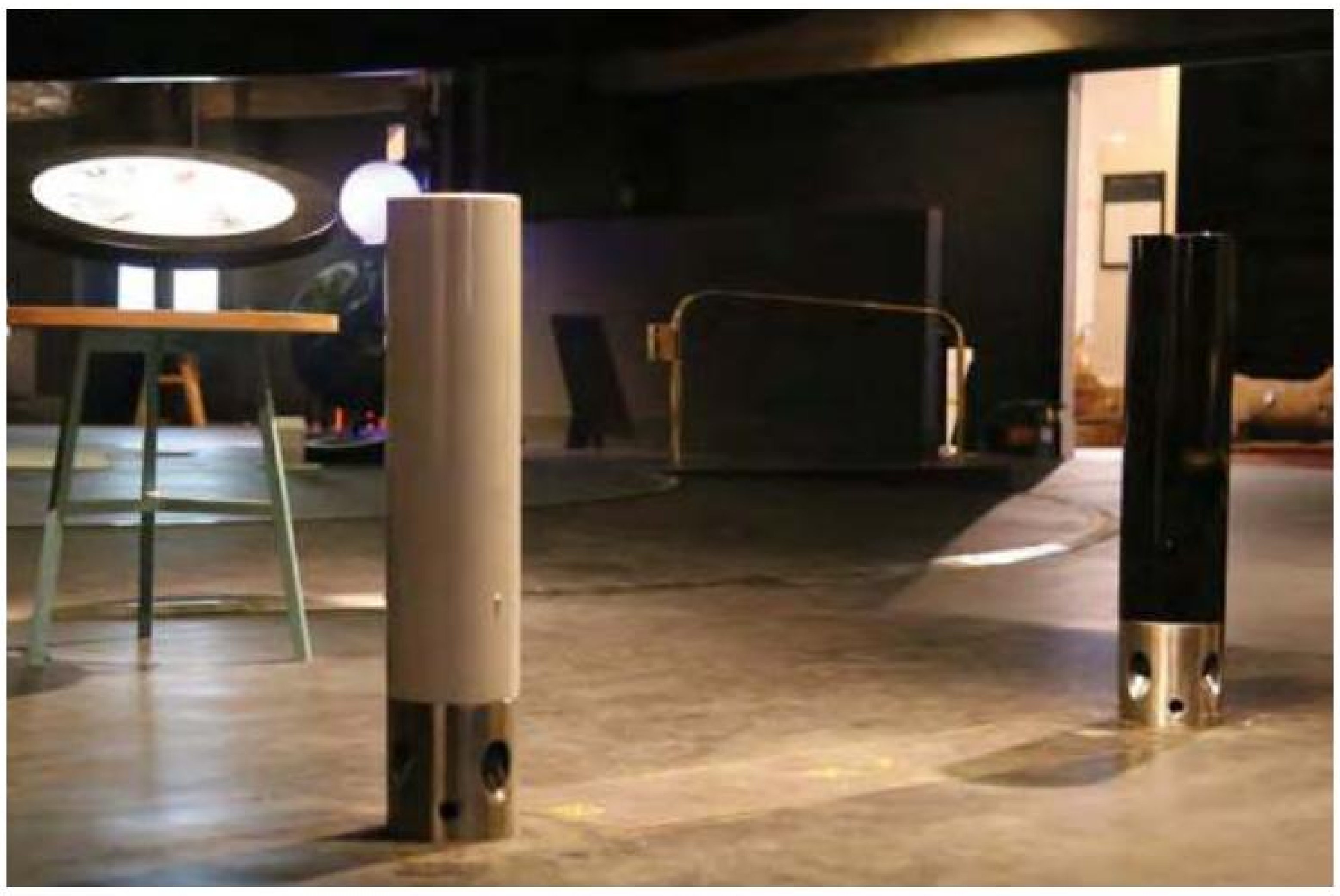


Fig.6. State of the art in Theremin design – the hybrid digital/analogue *Theremin Bollard* in the Science Museum, London. I designed and built the analogue front end for these instruments.

on a metal plate on damp ground.) The foot-to-floor capacitance is much more than the capacitance variation sensed by the plates, but because these two capacitances are in series the variable capacitance dominates and the system works well. Theremins give a wider note range if their signal ground is connected to a proper earth, such as mains earth, often achieved via the amplifier that the Theremin is plugged into. I sometimes put an earth terminal on Theremins, which has been very useful in some situations. I knew one player who used an earthing strap attached to his wrist. I had to clip a Theremin earth to a radiator in a school hall once because of its poor mains earth.

The sound

Typically, a Theremin's output is a distorted sine wave, as shown in Fig.4. The sinewave is relatively pure at higher frequencies around 1kHz, but becomes more distorted towards the bottom end, often becoming quite buzzy sounding near 100Hz. Acoustic instruments and the human voice do this naturally, so this effect enhances the musicality of the Theremin sound. This occurs because the shape of the sinewave becomes distorted as the oscillator frequencies get closer. This effect is called 'pulling', and eventually the two oscillators lock together at the same frequency. This point is called the 'null point', because there is then no audio output.

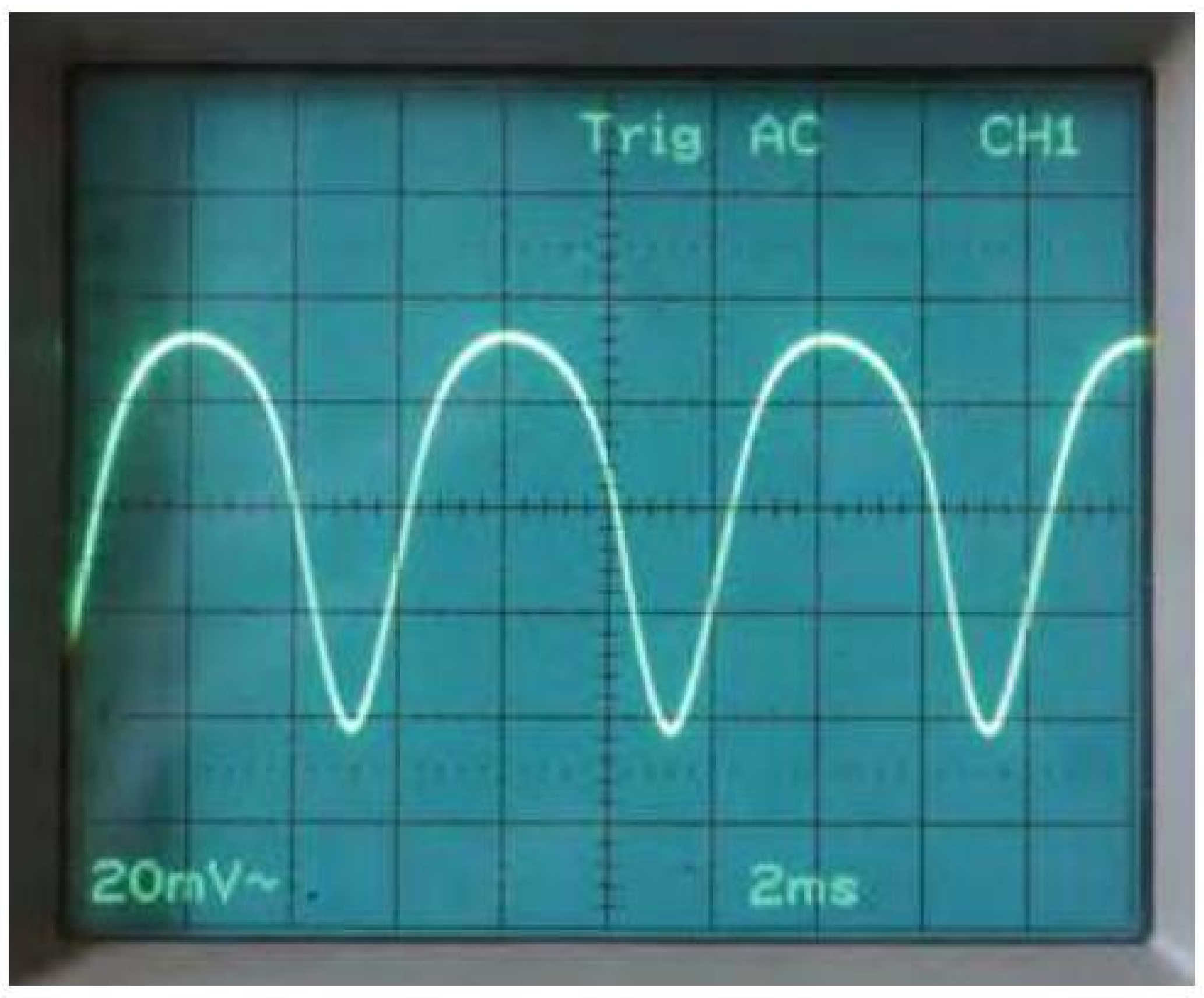


Fig.4. Characteristic distorted sinewave produced by a Theremin with a degree of locking at low frequency.

Processing

It helps greatly to add digital delay or spring-line reverb to the sound and I will be doing a dedicated delay board using the £0.50 Princeton PT2399 delay chip soon. This chip was used in the *Gen X-1* synthesiser and also the *Midi Ultimate Synthesiser* in *PE* (see May 2019). There is no need to add vibrato or tremolo; shaky hands add more than enough.

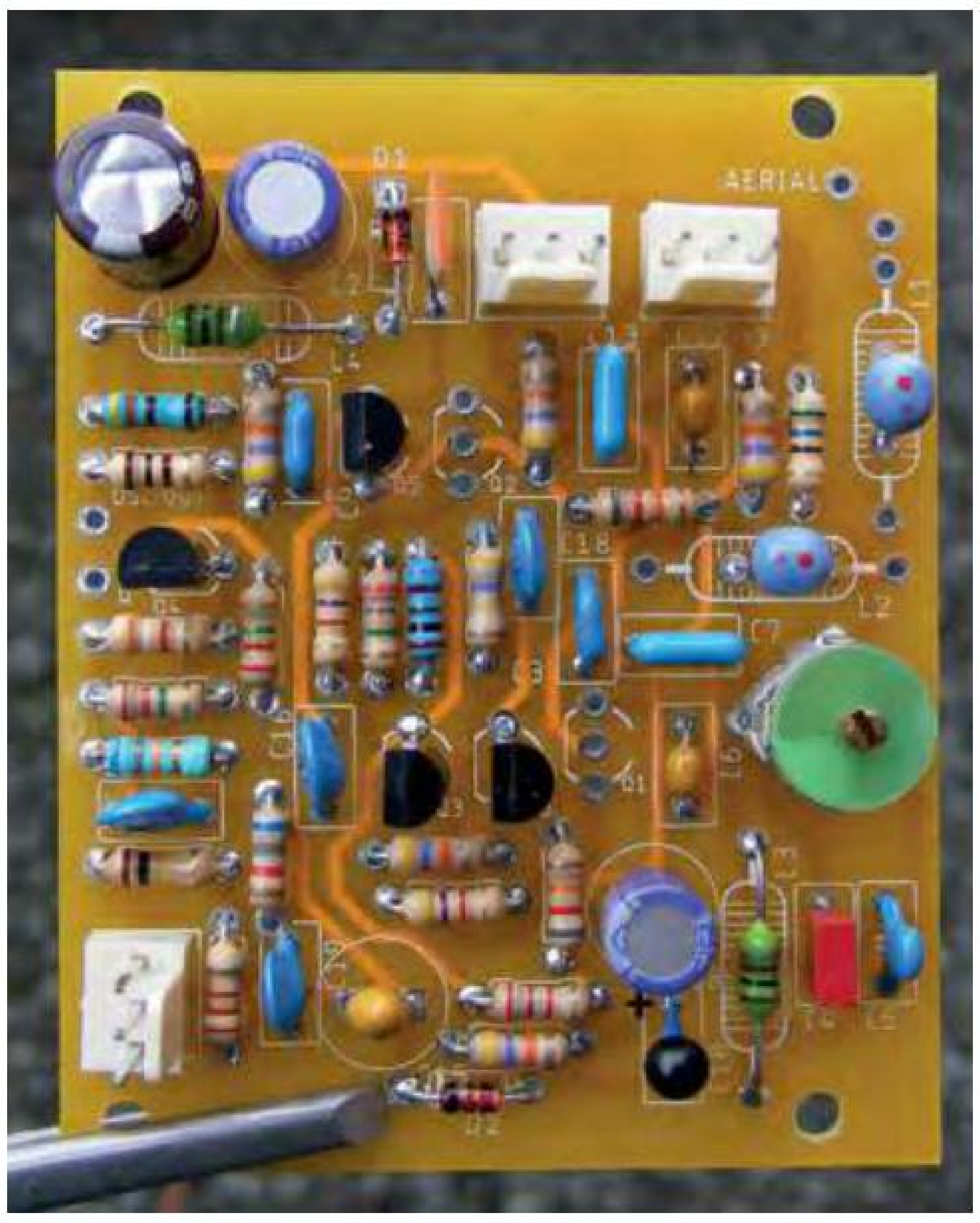


Fig.5. The pitch PCB – basically a simple pitch-only Theremin on one board.

Pitch-only Theremins

Simple Theremins offer only pitch control, and rely on the zero beat point to prevent the noise being continuous. My best seller has been the *Pocket Theremin*, which is still in production after 23 years. This was published in *EPE* in September 1995 and later in the *Modern Amateur Electronics Manual* (see supplement 55). The circuitry and component values have been refined over the years. This circuitry still forms the basis for the pitch module design shown in Fig.5. Theremin circuits can be a critical mixture of subtle radio frequency and audio design and can take years to refine. The technique is more 'design by soldering iron' rather than a simulator.

Playing the Theremin

Although it's much more accurate to play notes on a keyboard, there is a magical theatrical quality to playing the Theremin; having to stay deathly still, while moving one's hands in thin air, seemingly generating music out of nothing. There are, of course, other contactless gesture-based musical control systems, based on infrared, ultrasonics and microwaves, but they're not the

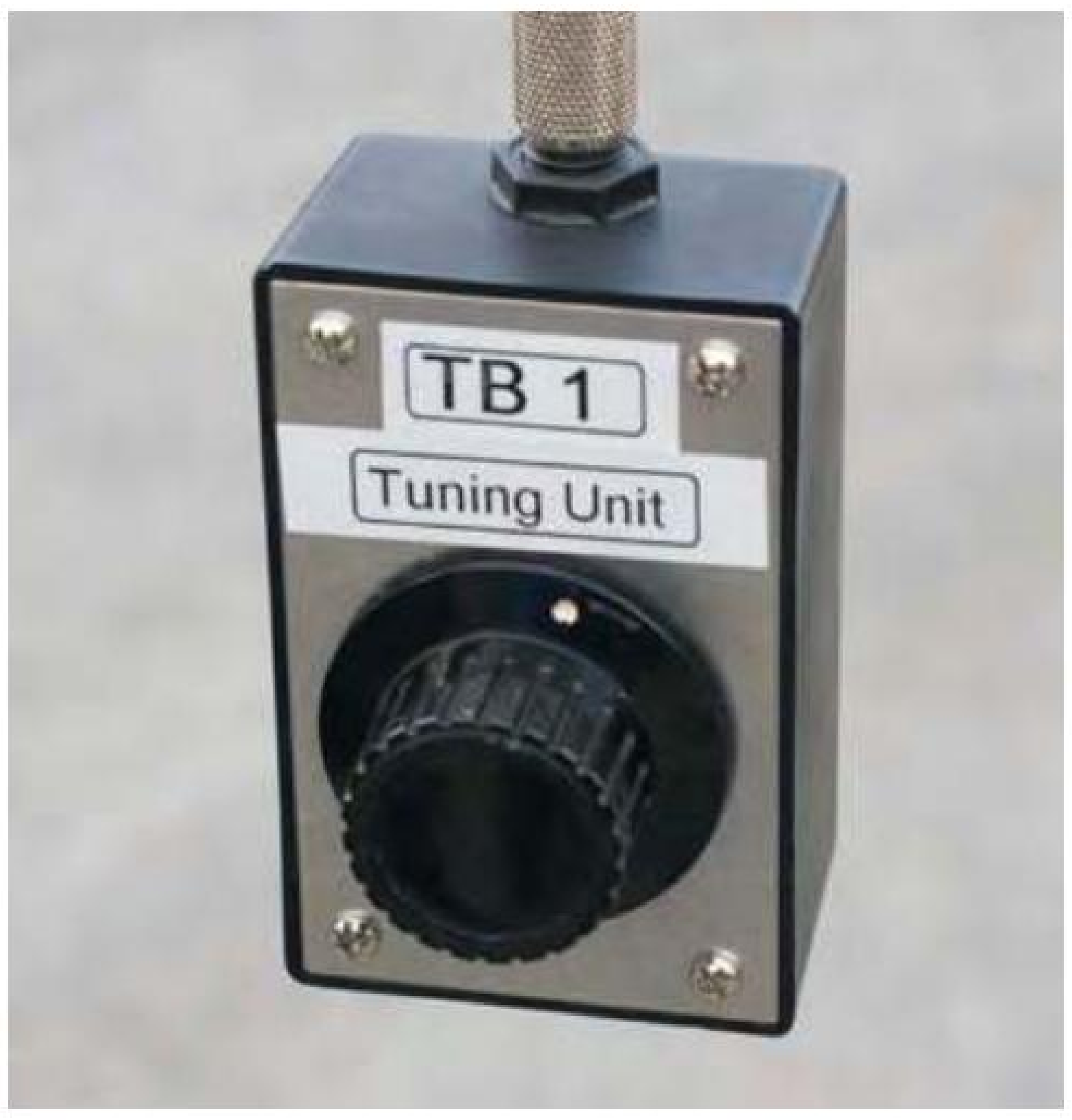


Fig.7. Putting the tuning pot in a (cable-linked) box displaced from from the aerial makes a Theremin much easier to tune.

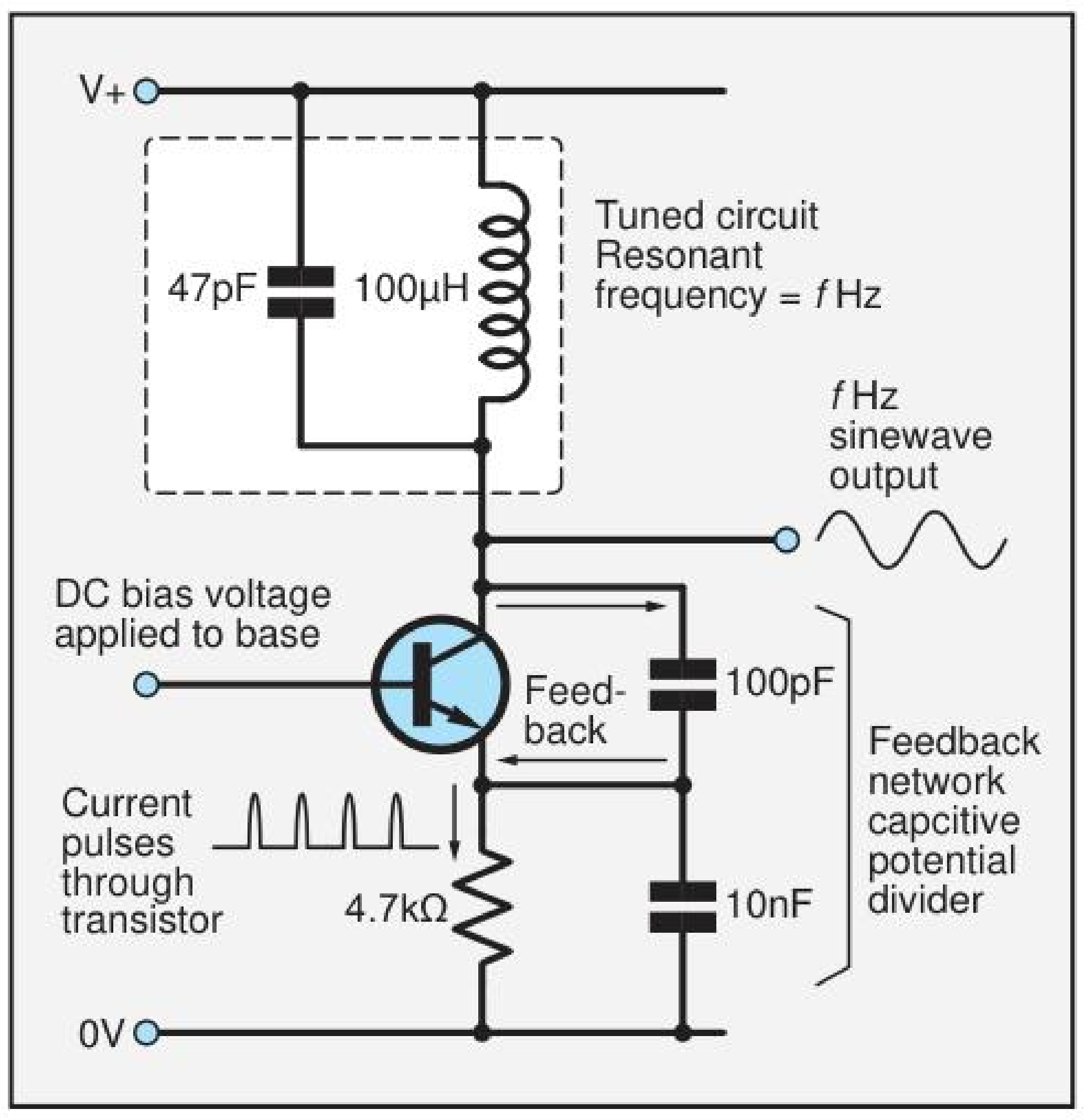

V+
Tuned circuit
Resonant
frequency = f Hz
47pF
100µH
f Hz
sinewave
output
DC bias voltage
applied to base
Feed-
back
100pF
Feedback
network
capcitive
potential
divider
Current
pulses
through
transistor
4.7kΩ
10nF
0V

Fig.8. Class-C oscillator operation.

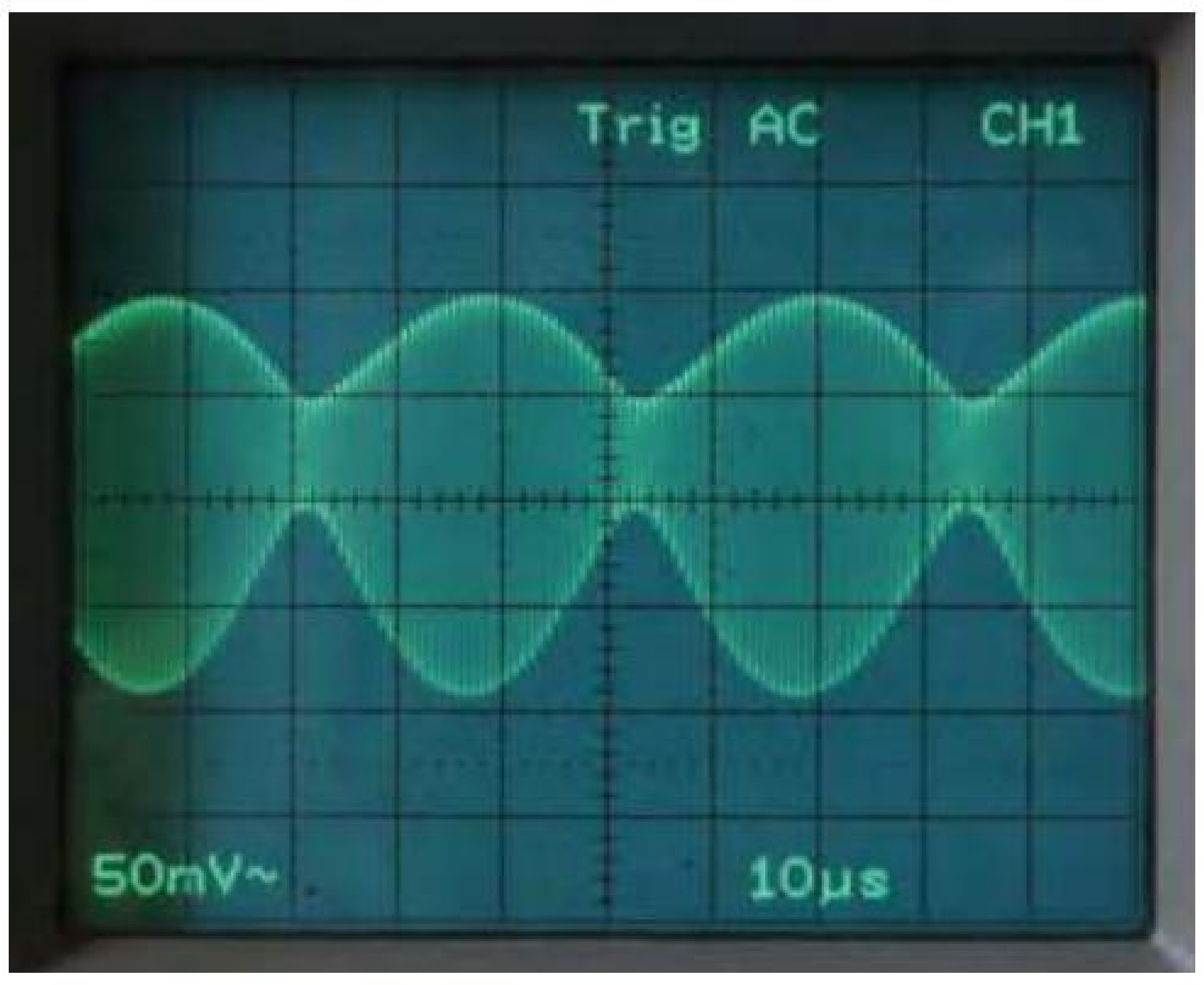

Trig AC
CH1
50mV~
10µs

Fig.9. Heterodyne waveform – the envelope is the beat frequency.

real deal. The Theremin has a unique omni-directional analogue response with no latency. When playing, I can shut my eyes and feel I'm pulling on some imaginary strings. Even now, the best approach is to use the original Theremin circuit topology with analogue-to-digital conversion and some clever digital signal processing, triggering stored musical sounds. This was used for the *Theremin Bollards* in the Wonderlab exhibition currently at the London Science Museum, conceived by David Young. (Fig.6). The public love them because the software removes all the 'bum notes', which are inevitable with inexperienced use of the Theremin. It normally takes years of practice to play in tune. The front-end circuitry was the same circuit used here, but with a remote tuning pot (Fig.7) and a pulse output waveform.

Theremin oscillators

Standard RF oscillator configurations are used for the oscillators. The Colpitts topology is preferred since it does not need a tapped inductor. The 'tapping' is provided by a pair of capacitors instead. This tapping provides the positive feedback for oscillation to take place. A good sinewave is produced by the high Q resonance of the tuned circuit. No complex amplitude control circuitry is needed for inductor/capacitor oscillators, unlike resistor/capacitor audio oscillators. The operation mode is normally class-C, where the active device is only on for less than half a cycle, which 'kicks' the resonant circuit causing it to ring continuously (see Fig.8). This mode gives us a great advantage, since varying the DC bias can control the width of the conduction

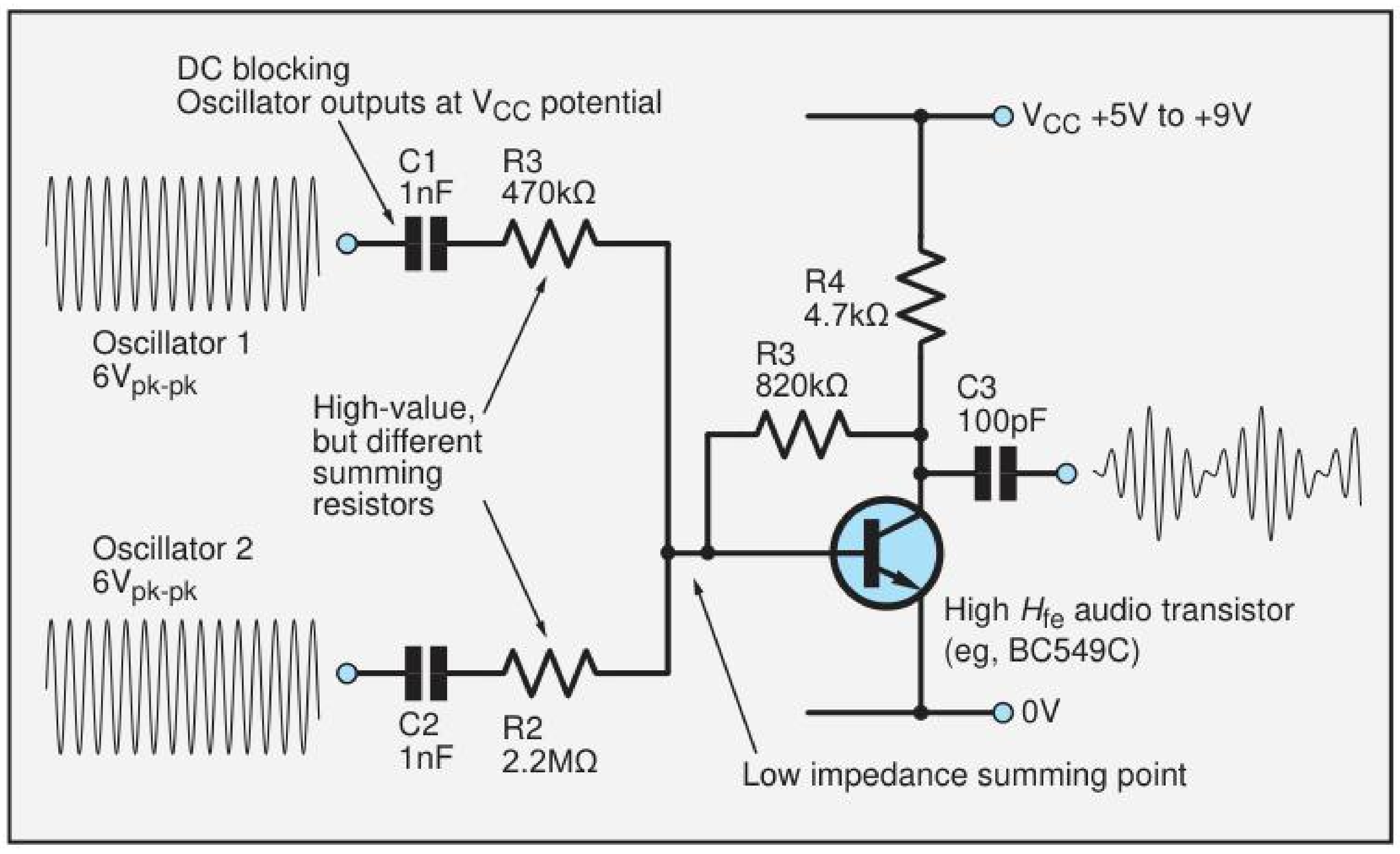

DC blocking
Oscillator outputs at V_{CC} potential
C1
1nF
R3
470kΩ
Oscillator 1
$6V_{pk-pk}$
High-value,
but different
summing
resistors
Oscillator 2
$6V_{pk-pk}$
C2
1nF
R2
2.2MΩ
V_{CC} +5V to +9V
R4
4.7kΩ
R3
820kΩ
C3
100pF
High H_{fe} audio transistor
(eg, BC549C)
0V
Low impedance summing point

Fig.10. A simple bipolar transistor mixer stage.

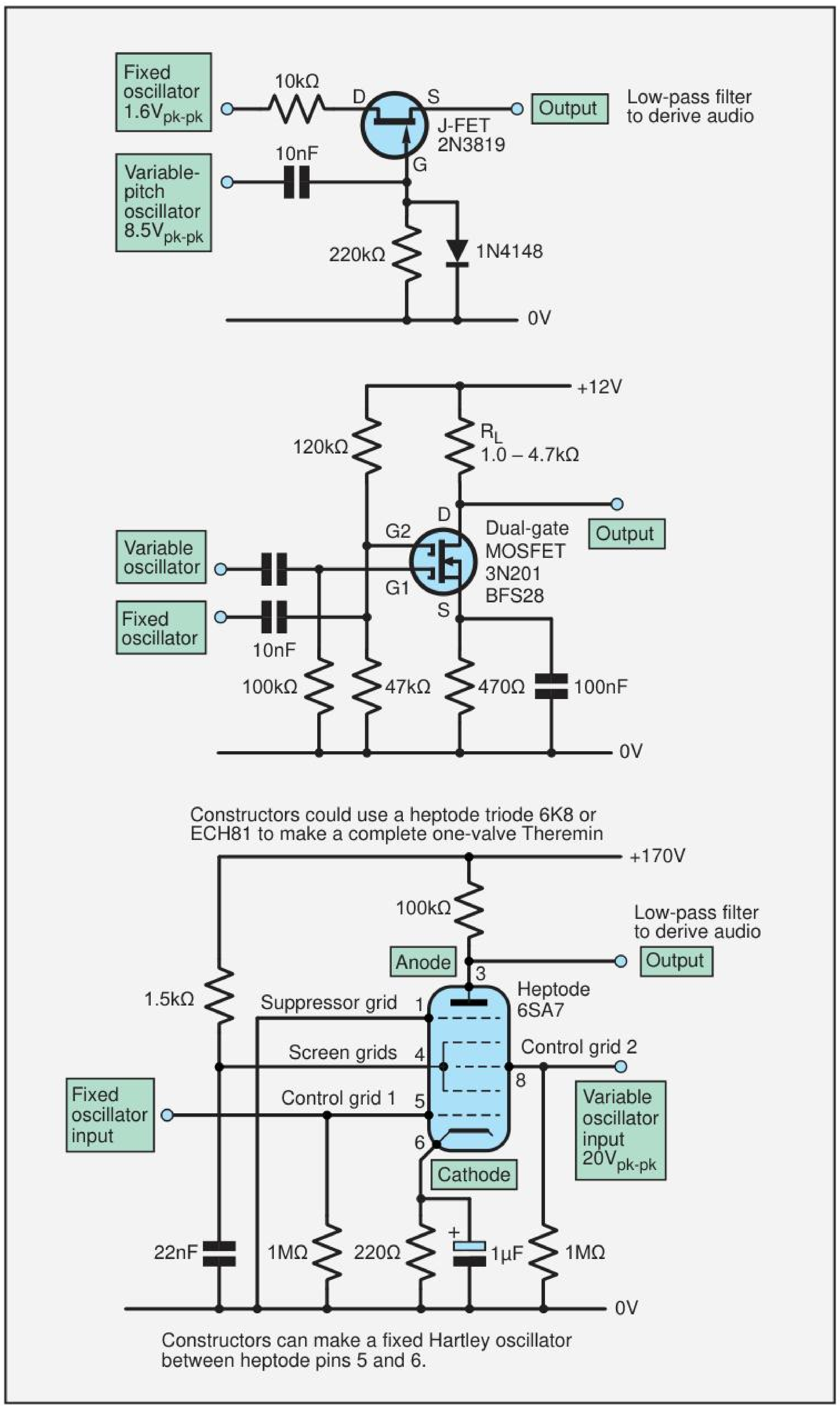

Fixed
oscillator
$1.6V_{pk-pk}$
10kΩ
D
S
J-FET
2N3819
Output
Low-pass filter
to derive audio
Variable-
pitch
oscillator
$8.5V_{pk-pk}$
10nF
G
220kΩ
1N4148
0V
+12V
120kΩ
R_L
1.0 – 4.7kΩ
D
G2
Dual-gate
MOSFET
3N201
BFS28
Output
Variable
oscillator
G1
S
Fixed
oscillator
10nF
100kΩ
47kΩ
470Ω
100nF
0V
Constructors could use a heptode triode 6K8 or
ECH81 to make a complete one-valve Theremin
+170V
100kΩ
Low-pass filter
to derive audio
Anode
3
Output
1.5kΩ
Suppressor grid
1
Heptode
6SA7
Screen grids
4
Control grid 2
8
Fixed
oscillator
input
Control grid 1
5
Variable
oscillator
input
$20V_{pk-pk}$
6
Cathode
22nF
1MΩ
220Ω
+
1µF
1MΩ
0V
Constructors can make a fixed Hartley oscillator
between heptode pins 5 and 6.

Fig.11. Top-down: Mixers – J-FET, dual-gate MOSFET, heptode.

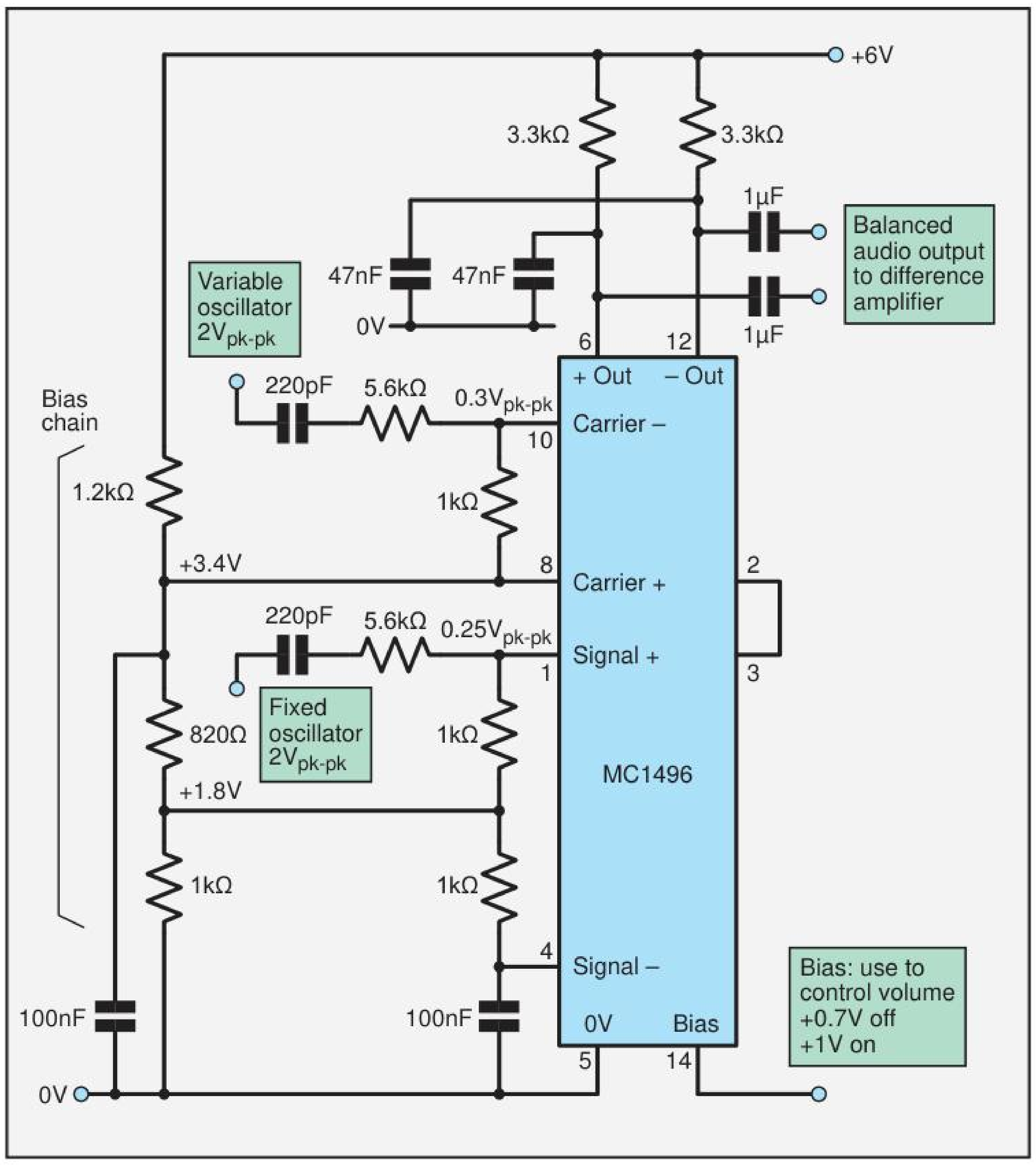

+6V
3.3kΩ
3.3kΩ
1µF
Balanced
audio output
to difference
amplifier
1µF
Variable
oscillator
$2V_{pk-pk}$
47nF
47nF
0V
6
12
+ Out
– Out
Bias
chain
220pF
5.6kΩ
$0.3V_{pk-pk}$
10
Carrier –
1.2kΩ
1kΩ
+3.4V
8
Carrier +
2
220pF
5.6kΩ
$0.25V_{pk-pk}$
Signal +
1
3
Fixed
oscillator
$2V_{pk-pk}$
820Ω
1kΩ
MC1496
+1.8V
1kΩ
1kΩ
4
Signal –
100nF
100nF
0V
Bias
5
14
Bias: use to
control volume
+0.7V off
+1V on
0V

Fig.12. The MC1496 double-balanced mixer gives excellent waveform purity.

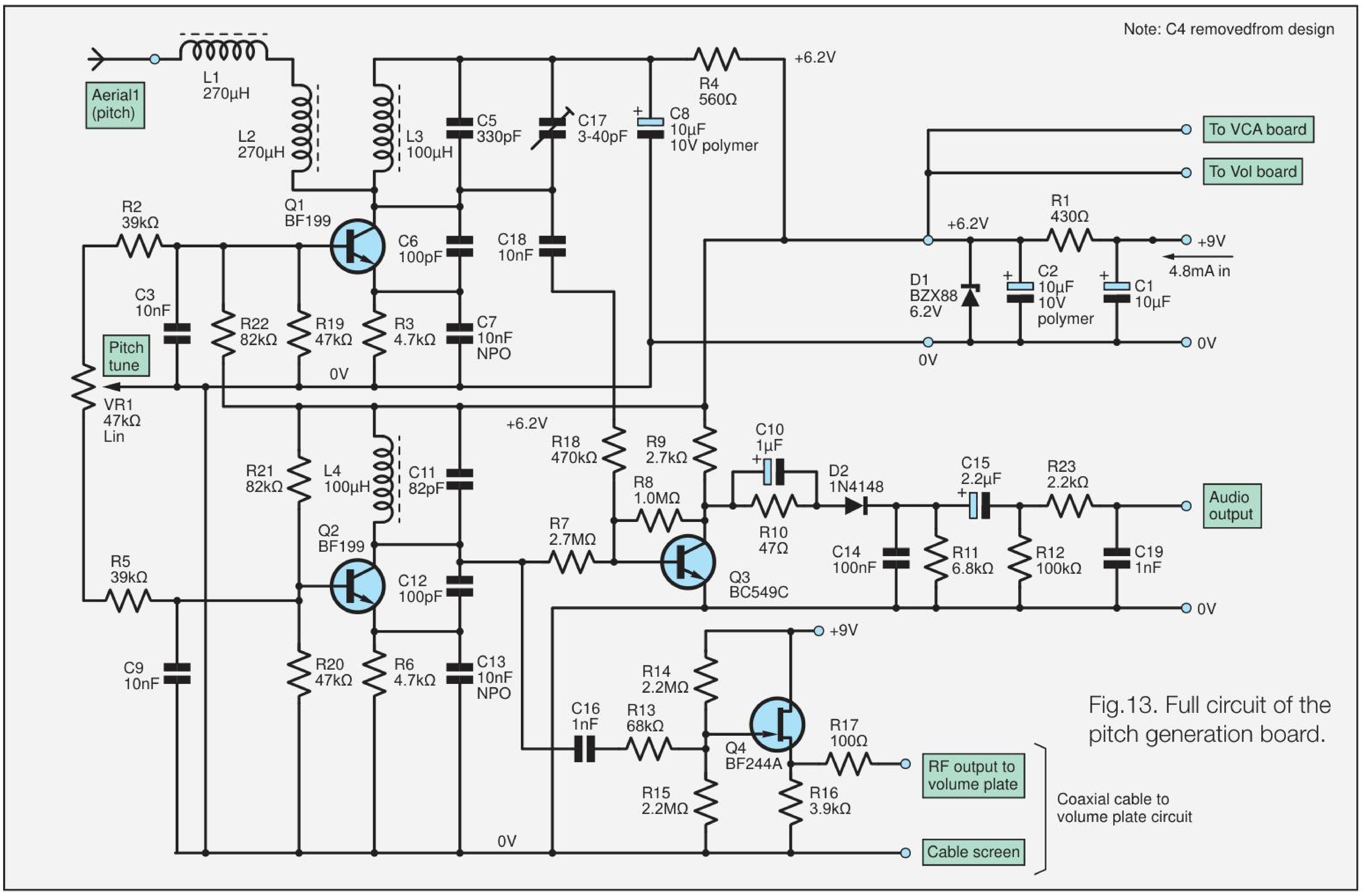

Note: C4 removedfrom design
Aerial1 (pitch)
L1 270µH
L2 270µH
L3 100µH
C5 330pF
C17 3-40pF
C8 10µF 10V polymer
R4 560Ω
+6.2V
To VCA board
To Vol board
R2 39kΩ
Q1 BF199
C6 100pF
C18 10nF
R1 430Ω
+6.2V
+9V
4.8mA in
D1 BZX88 6.2V
C2 10µF 10V polymer
C1 10µF
0V
C3 10nF
Pitch tune
R22 82kΩ
R19 47kΩ
R3 4.7kΩ
C7 10nF NPO
0V
VR1 47kΩ Lin
+6.2V
R18 470kΩ
R9 2.7kΩ
C10 1µF
D2 1N4148
C15 2.2µF
R23 2.2kΩ
Audio output
R21 82kΩ
L4 100µH
C11 82pF
R8 1.0MΩ
R10 47Ω
Q2 BF199
R7 2.7MΩ
C14 100nF
R11 6.8kΩ
R12 100kΩ
C19 1nF
R5 39kΩ
C12 100pF
Q3 BC549C
0V
+9V
C9 10nF
R20 47kΩ
R6 4.7kΩ
C13 10nF NPO
R14 2.2MΩ
C16 1nF
R13 68kΩ
Q4 BF244A
R17 100Ω
RF output to volume plate
R15 2.2MΩ
R16 3.9kΩ
Coaxial cable to volume plate circuit
0V
Cable screen

Fig.13. Full circuit of the pitch generation board.

period. This gives an effective way of providing a small degree of voltage control of the frequency, which means a simple potentiometer can provide the fine tuning control so necessary with Theremins. I then developed this circuit to turn the bias of one oscillator down while that of the other up. This gave me extra tuning range and kept the circuit symmetrical to minimise drift. Out went the expensive variable inductors, capacitors and varicap diodes I originally used. This gave me a considerable cost advantage. I didn't patent it because Theremins are never going to generate much revenue. Big companies won't touch this truculent technology because they are unstable and need tricky adjustments to produce.

The mixer/demodulator stage

The outputs of the oscillators have to be summed together in a mixer stage. The resulting output is an amplitude-modulated signal just the same as in AM radio, with sidebands or sum and difference frequencies. The carrier frequency is the RF oscillator frequency and the amplitude modulation is the audio beat frequency we need to extract. This waveform is shown in Fig.9. Referring to the pitch board circuit in Fig.13, we can use the radio technique of a detector diode (Fig.10) to do this. The output is then low-pass filtered by capacitors C14 and C19, leaving the difference frequency. In this case, the mixer uses a bipolar transistor (TR3) with resistive mixing via R18 and R7. The level of one oscillator is higher than the other. This is to avoid distortion due to the envelope completely dipping to zero.

A FET can also be used – one oscillator chops the other, see Fig.11. This gives lower noise but offers no real advantage since a bit of noise enhances the Theremin sound. A dual-gate MOSFET gives good isolation between the oscillators; better still is the heptode valve, which has intervening screen grids.

The most sophisticated system uses a ring modulator circuit based on the MC1496, see Fig.12. When set up for the right levels of less than $250mV_{pk-pk}$, this can give a very low distortion sinewave output. This was the system John Clark used in his *Silicon Chip* August 2000 design (plus vital overdrive). Although this technique is useful in function generators, a pure sinewave sounds a bit boring for musical use.

The final circuit for the pitch board is shown in Fig.13. This can be used on its own as a pitch-only Theremin, such as the design shown in Fig.14. Playing a pitch-only Theremin requires a special technique. The tuning has to be set so that it is at zero beat or at its null point when the hand is away from the aerial.

Next month

In *Part 2* next month we will construct the first *PE Theremin* sub-system – the pitch board shown in Fig.5 and Fig.13.

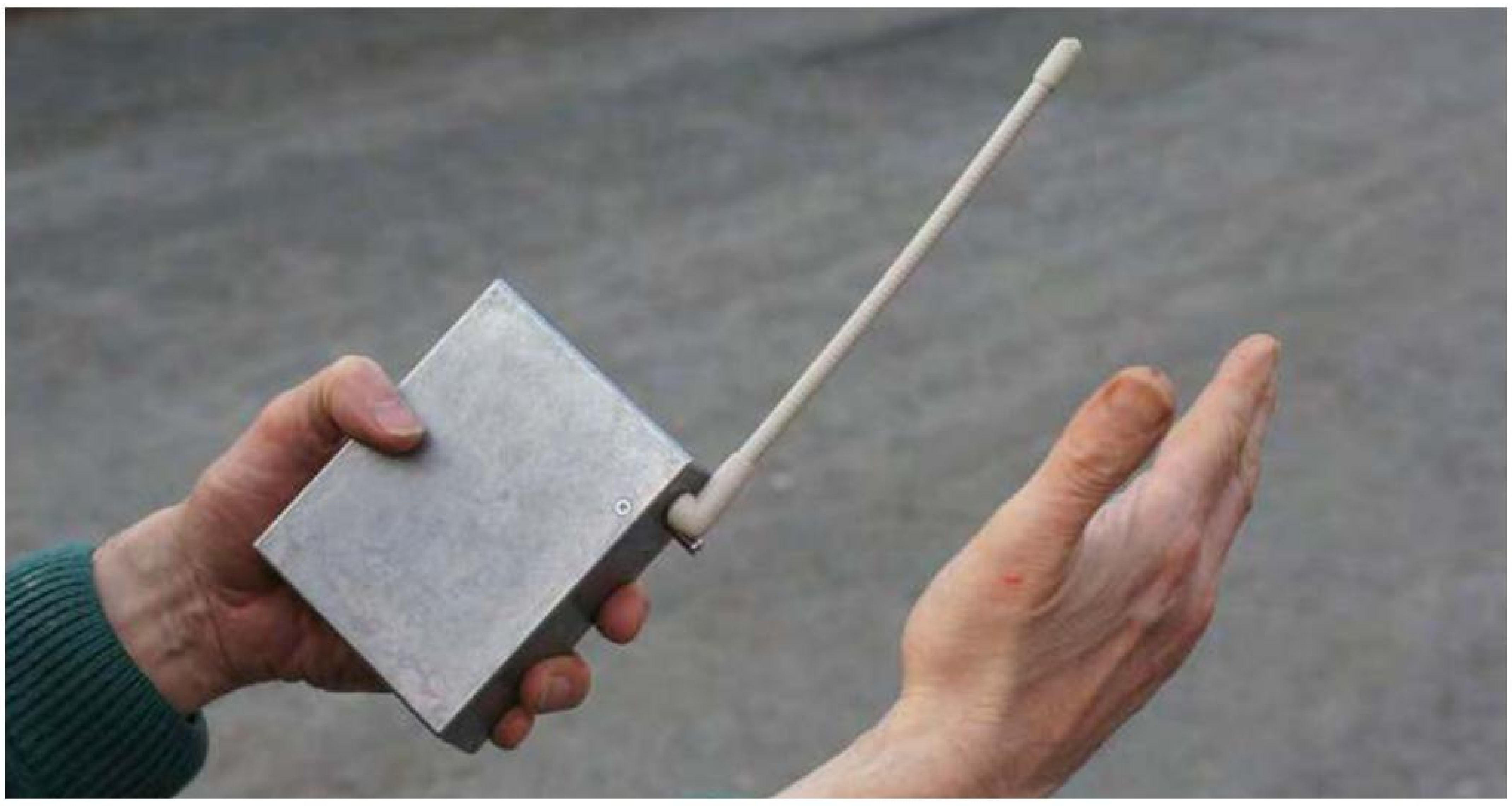
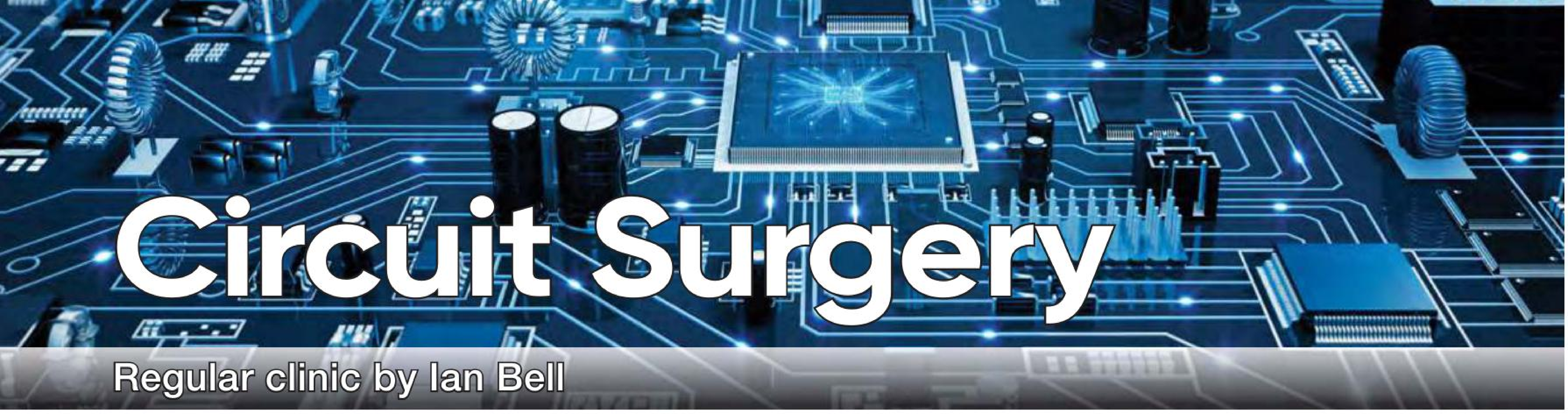


Fig.14. Pitch-only Theremin based on the pitch board.



Circuit Surgery

Regular clinic by Ian Bell

Introduction to Circuit Simulation with LTspice – Part 5

Last month, in the article on comparators, LTspice was used to provide some example waveforms, but we did not go into the details of how these simulations were set up – that is what we will be looking at this month. We covered some basics of SPICE analogue circuit simulation, and LTspice in particular, in a recent series of articles (*EPE*, October 2018 to January 2019) and this month's *Circuit Surgery* is a continuation of that series.

The general theme this time is one of modelling circuits that are not immediately available as basic circuit elements, or as components in the library of devices downloaded with LTspice – for example, last month we used models of the LM741 op amp and LM393 comparator. Models of such devices can often be downloaded from manufacturers' websites and included in simulation, and they can be added to the library if you want. It is also possible to create your own models of devices such as op amps. We will show how to create a very basic model as this helps illustrate how the process works – the use of downloaded models follows from this.

Hierarchical schematics

Before looking at how we model devices it is helpful to understand how hierarchical schematics are handled in LTspice – this is the use of existing circuits within other schematics. The main schematic is called the 'top-level circuit' and the smaller circuits within it are 'subcircuits', which are represented as symbols on the top-level schematic. This is illustrated in Fig.1, which shows an RC circuit defined as a subcircuit for which a symbol has been created and then used in a top-level schematic to create a three-stage RC filter. Hierarchical schematics can be built over many levels of circuits-within-circuits if required.

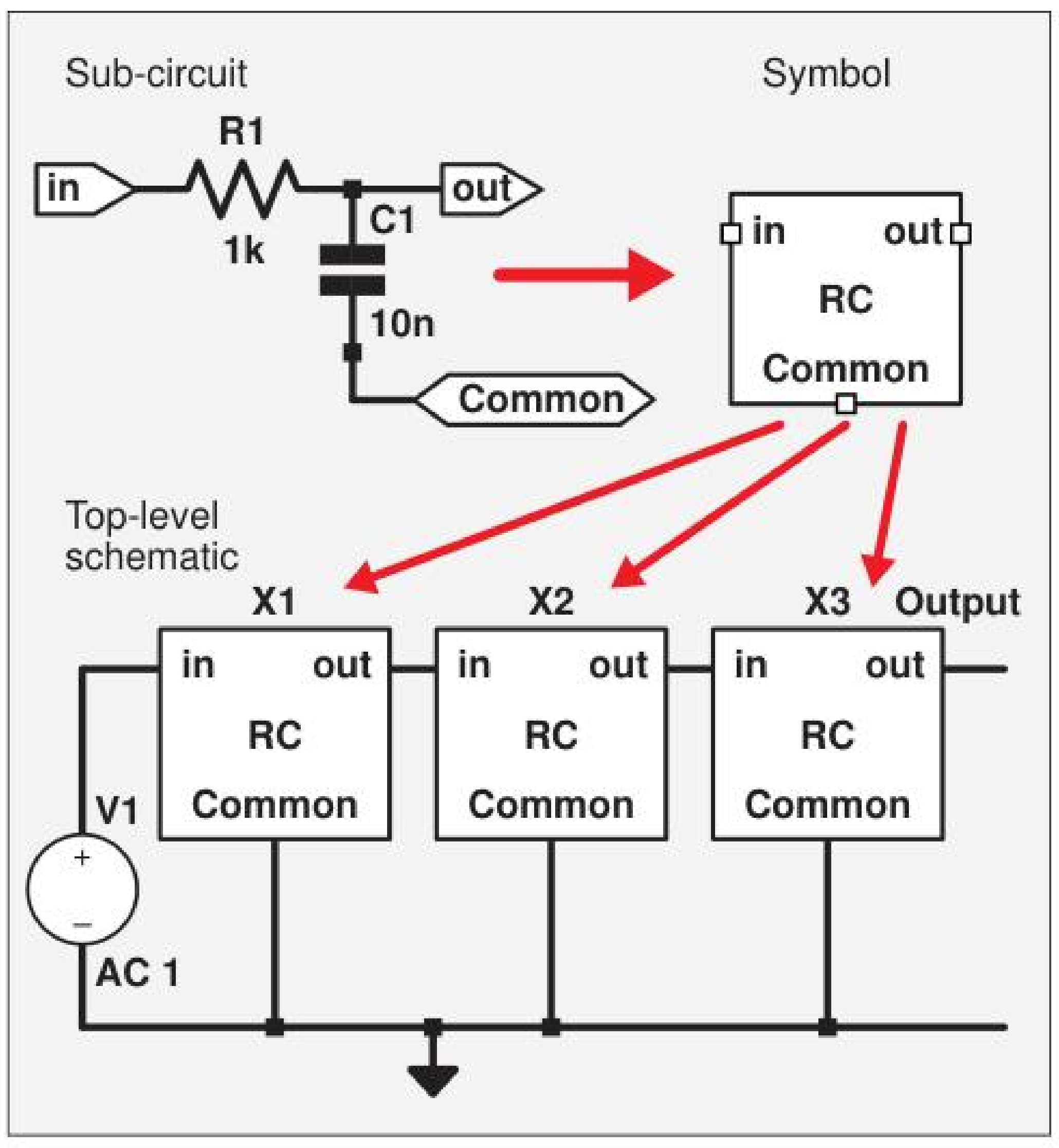


Fig.1. Example of subcircuit, symbol and top-level schematic.

The subcircuit's symbol has to be defined (drawn and saved somewhere) before it can be used – LTspice has a tool specifically for drawing symbols. The default subcircuit symbol is simply a rectangle, but other shapes can be drawn. In SPICE, a component type (element) is identified by its initial letter (eg, voltage sources: `V`, resistors: `R` and capacitors: `C`). Subcircuits have the initial letter `X`.

Macromodels

It is possible to create relatively simple circuits in SPICE, which are electrically equivalent, at least to some extent, to original circuits containing many components, such as an op amp. These circuits are known as 'macromodels'. The simplest versions can be used as generic versions of components such as op amps and potentially provide faster simulation than using a full schematic. More sophisticated macromodels can mimic the behaviour of specific real devices. These are used by device manufacturers to enable users to simulate designs containing their components without having to publish the actual schematics or implementation details.

The most basic macromodels are built using just passive components such as resistors and capacitors, together with controlled sources (other models may use some transistors for accuracy, for example, just for the input stage). Controlled sources are like the voltage or current sources used in SPICE for signals and power supplies, but the output is determined by another voltage or current in the circuit. There are four types:

`E`	voltage-controlled voltage source
`F`	current-controlled current source
`G`	voltage-controlled current source
`H`	current-controlled voltage source

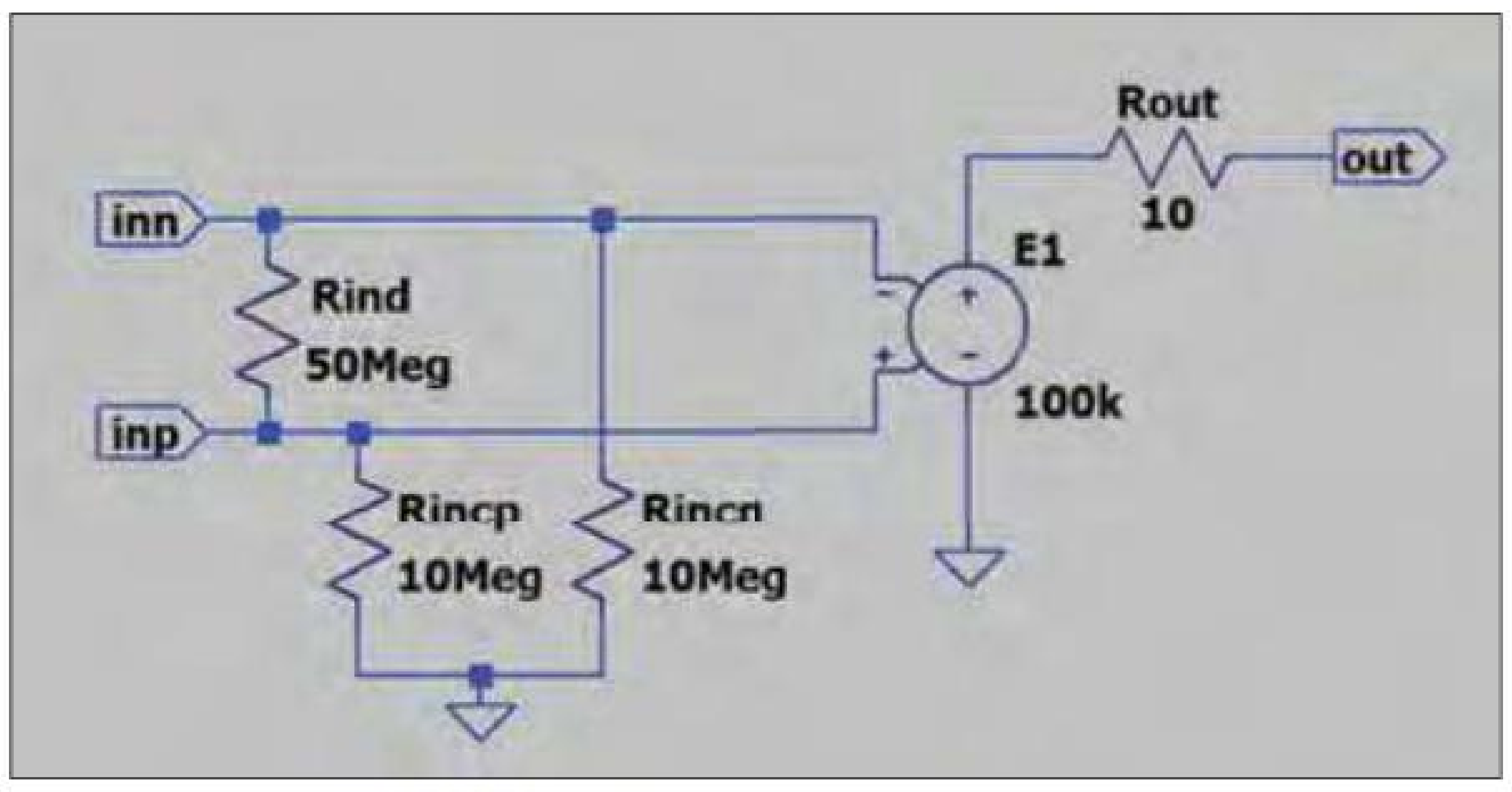


Fig.2. Simple op amp model in LTspice.

`E`, `F`, `G` and `H` are the initial letters of the element names of these types of component in SPICE.

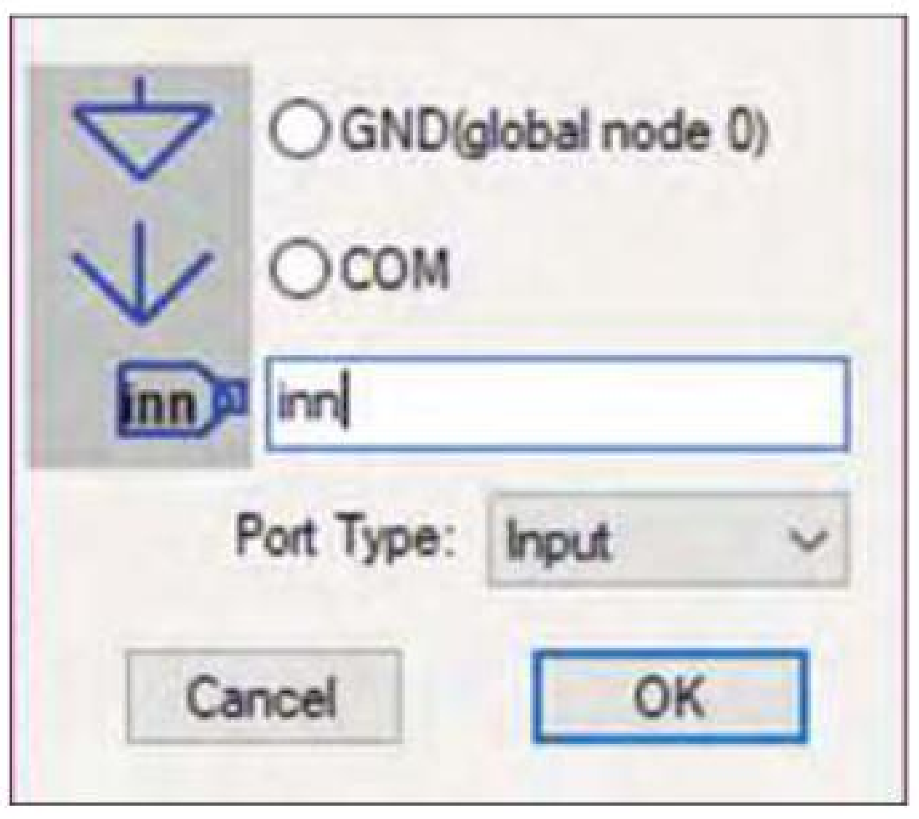
To define the behaviour of a controlled source it is necessary to define the relationship (transfer function) between the controlling voltage or current and the source's output voltage or current. The simplest version is a constant gain – the output is the input value multiplied by the gain. Other approaches are available, including piecewise linear, where a look-up table of pairs of numbers specifies the transfer function. Further flexibility is provided by the behavioural sources (`BV` and `BI` elements) which allow the use of mathematical expressions to define the input-output relationship. A source of this type was used, with a random number function, to create the noisy-waveform comparator simulation last month. For full details on these sources, consult the LTspice help.

Amplifier model

Fig.2 shows a simple macromodel of a high-gain differential-input amplifier – it is a nearly ideal op amp. We will use this to illustrate how to create hierarchical schematics and place subcircuits in libraries. The circuit in Fig.2 uses a voltage-controlled voltage source (`E1`) to produce an output voltage 100,000-times larger than the voltage difference between the `inp` and `inn` wires (amplifier inputs). Large gains such as this are typical of real op

Simulation files

The LTSpice files discussed in *Circuit Surgery* are available for download from the *PE* website.


○ GND(global node 0)

○ COM

● inn

Port Type: Input

Fig.3. Dialog for setting net names and port directions.

amps. The circuit uses large-value resistors to model both common-mode (`Rincp` and `Rincn`) and differential (`Rind`) input impedance. This is a simplification compared with a real op amp, which would also have capacitance as part of its input impedances. The circuit also has output impedance modelled using the resistor `Rout`.

Other major simplifications compared with real devices include the fact that the circuit has no limit on its output voltage – unlike real op amps, which are limited to their supply range, or lower. This could be added, for example, by including diode clamps, but our aim here is to look at how to work with circuits like this, rather than go into the details of creating them.

The schematic in Fig.2 is drawn in LTspice in the usual way, but has a feature which may not be familiar – the input/output pins. To include these, use the Label Net button, or right click on the wire and select Label Net, then set the port name to Input, Output or Bi-Direct as required, click OK and place the port symbol. The ports do not appear in the netlist (the code used as the description of the circuit to the simulator), they just make the schematic more readable. The pin direction does not affect the electrical behaviour of the simulation.

Creating a circuit symbol

To create a symbol manually, open the symbol editor using `File > New Symbol`. The symbol editor is like a simple drawing package – from the Draw menu it is possible to select options to draw rectangles, lines, arcs, circles and text. For example, a rectangle is shown in Fig.4. These items are just shapes in the component symbol – they are not interpreted in any way electrically by the simulator. Items in the drawing can be manipulated via the Edit menu, allowing you to move, drag, rotate and delete them. The small red circles are anchor points, which can be used to resize objects. The anchor points can be toggled on and off via the View menu. The crosshair is a reference location for when the symbol is placed on a schematic.

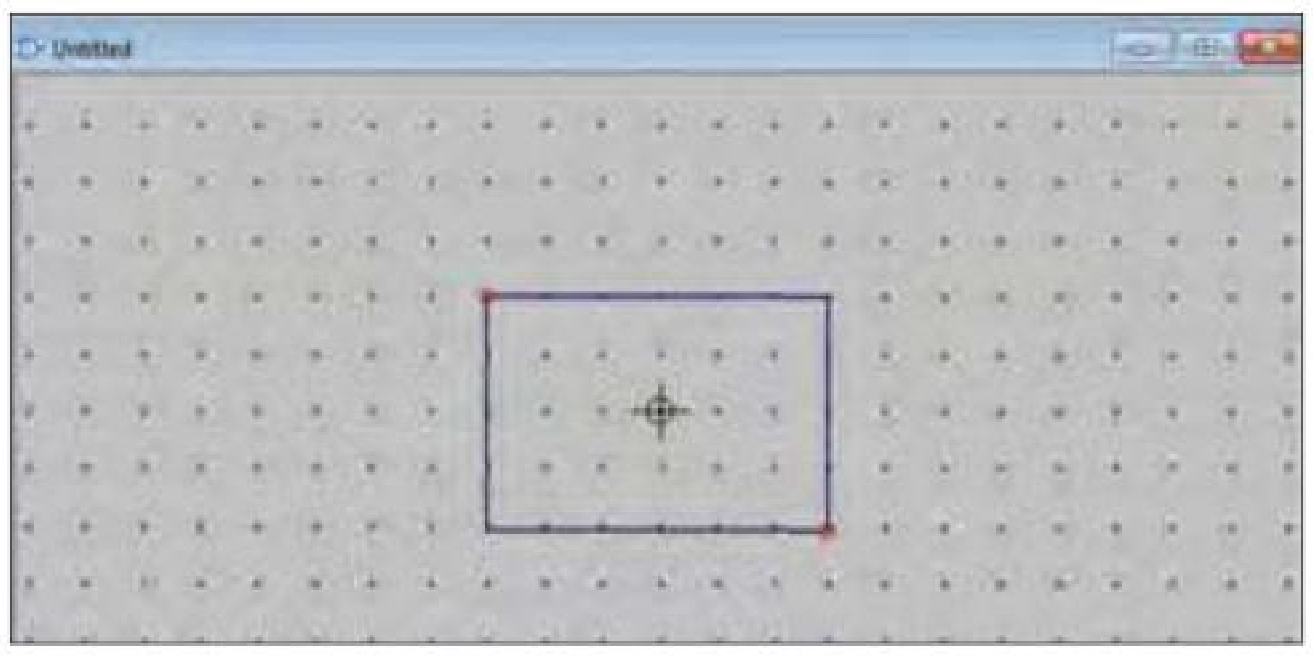
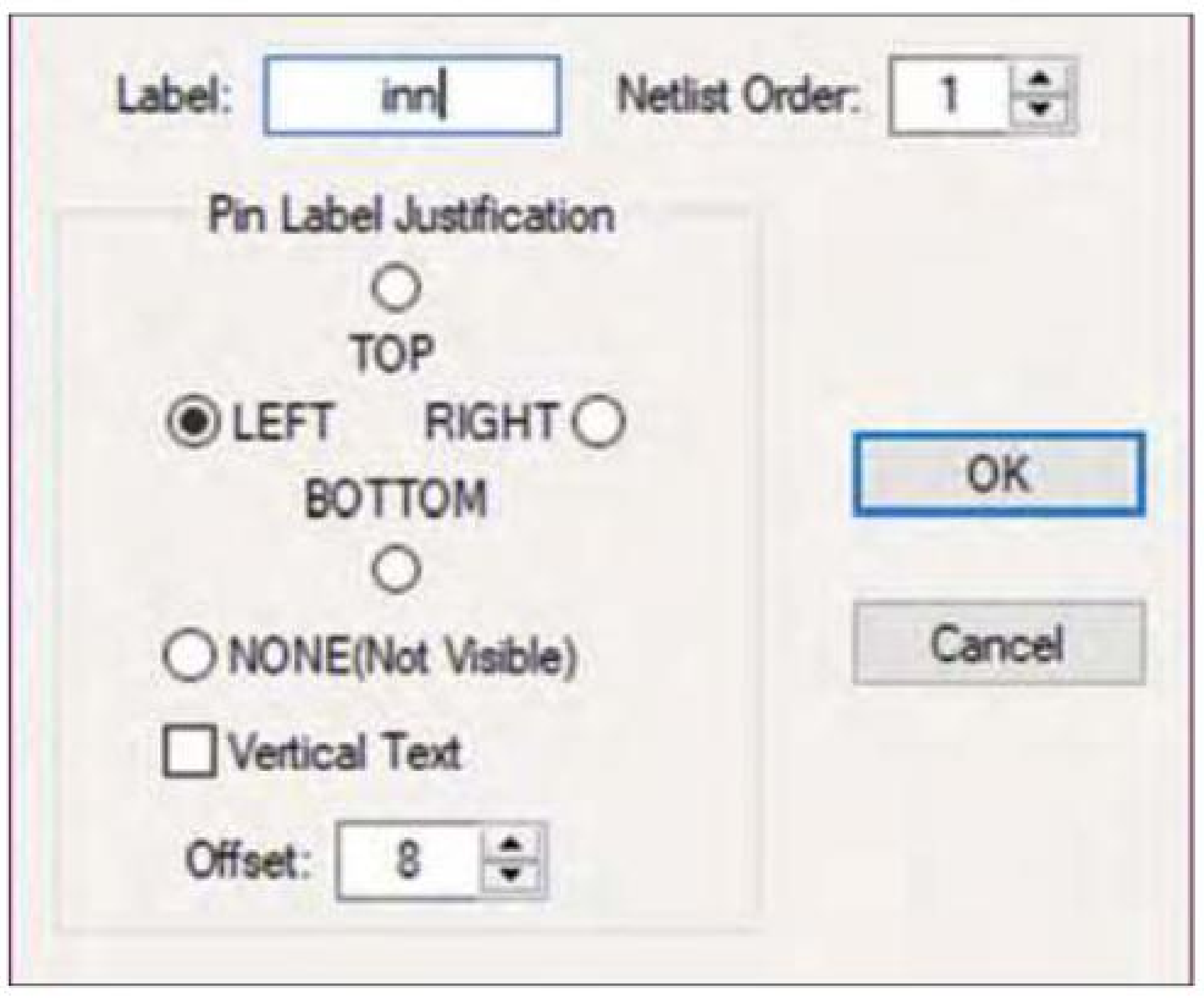


Fig.4. Example symbol drawing window showing a rectangle with its anchor points.


Label: inn

Netlist Order: 1

Pin Label Justification

○ TOP

● LEFT

○ RIGHT

○ BOTTOM

○ NONE(Not Visible)

☐ Vertical Text

Offset: 8

Fig.5. Pin Properties dialog in the LTspice symbol editor.

In order for the symbol to be used in a schematic its input and output pins must be defined. Pins are added to the symbol using `Edit > Add Pin/Port`. This opens a dialog allowing you to set up the pin – give it a name and choose how that name is displayed on the symbol (see Fig.5). When drawing a symbol for a hierarchical schematic the pin names must match the corresponding net names in the subcircuit schematic. For example, when creating a symbol for Fig.2 there must be three pins named `inn`, `inp` and `out` in the symbol; however, the names do not have to be displayed on the symbol. Note that the symbol pins do not have a direction (in, out) specified.

The netlist order must be set for each pin. Recall from *Part 2* of this series that SPICE circuits are described in text files called 'netlists'. Each component has a line of code to describe how it is connected, what parameter values it has, or which model it uses. For example, a diode using the 1N914 model connected to wires (nets) named `N1` and `N2`, may appear in the netlist as: `D1 N1 N2 1N914`

The order of `N1` and `N2` matter – the first wire listed (`N1` in this case) is the connection to the diode's anode and the second the cathode. The netlist order of connections of primitive elements is fixed, but when a new symbol is defined the order in which the pins occur must be specified – this is the purpose of the Netlist Order dropdown box in the Pin/Port Properties dialog.

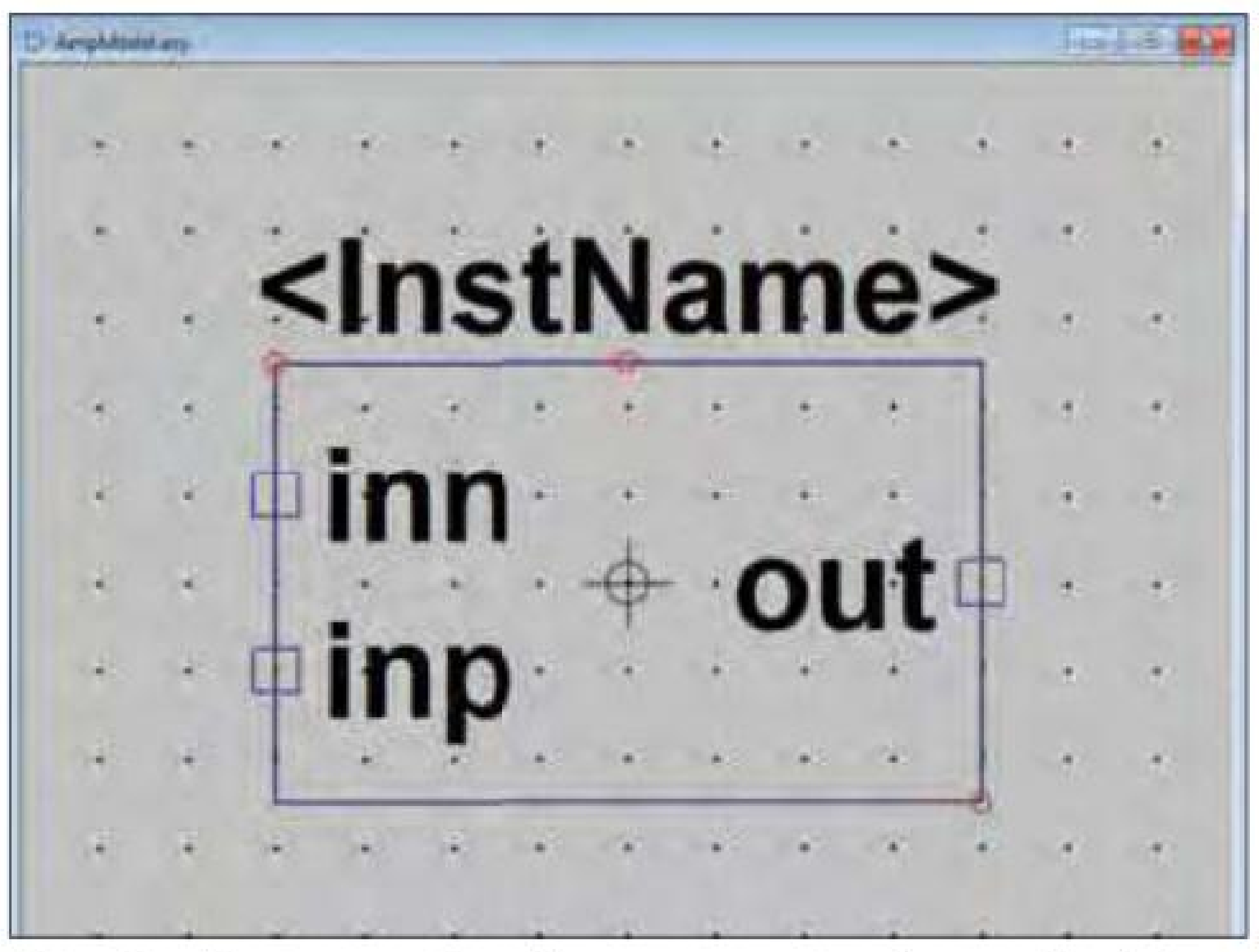


Fig.6. Automatically generated symbol for the circuit in Fig.2.

LTspice can generate symbols automatically for you. To do this, select the schematic and do `Hierarchy > Open this Sheet's Symbo`. The result of applying this to Fig.2 is shown in Fig.6. The small blue squares next to the pin names are the points on the symbols to which it will be possible to connect wires when the symbol is used in a schematic.

A third option for creating a symbol is to copy an existing symbol file and reuse it. The symbol files can be obtained from the libraries installed by LTspice. Symbol files for hierarchical subcircuits (extension `.asy`) must have the same name as the subcircuit schematic (extension `.asc`), so a copied file must be renamed to match the schematic file. The pin names in the symbol may also have to be edited to match the schematic.

Subcircuits and netlists

As just mentioned, SPICE circuits are actually input to the simulator as netlist text files. LTspice can work directly from netlist files and does not need a schematic. This is usually the case when working with downloaded models of devices, such as the op amp and comparator in last month's article. A basic knowledge of how subcircuit netlist files work is helpful when using downloaded models, or if you want to build your own library of subcircuits.

When creating hierarchical circuits the definition of a subcircuit requires a particular syntax in the netlist. This involves two SPICE directives (commands): `.subckt` and `.ends` (recall that SPICE directives always start with a dot). An example subcircuit netlist is shown below – this is for the RC circuit in Fig.1.

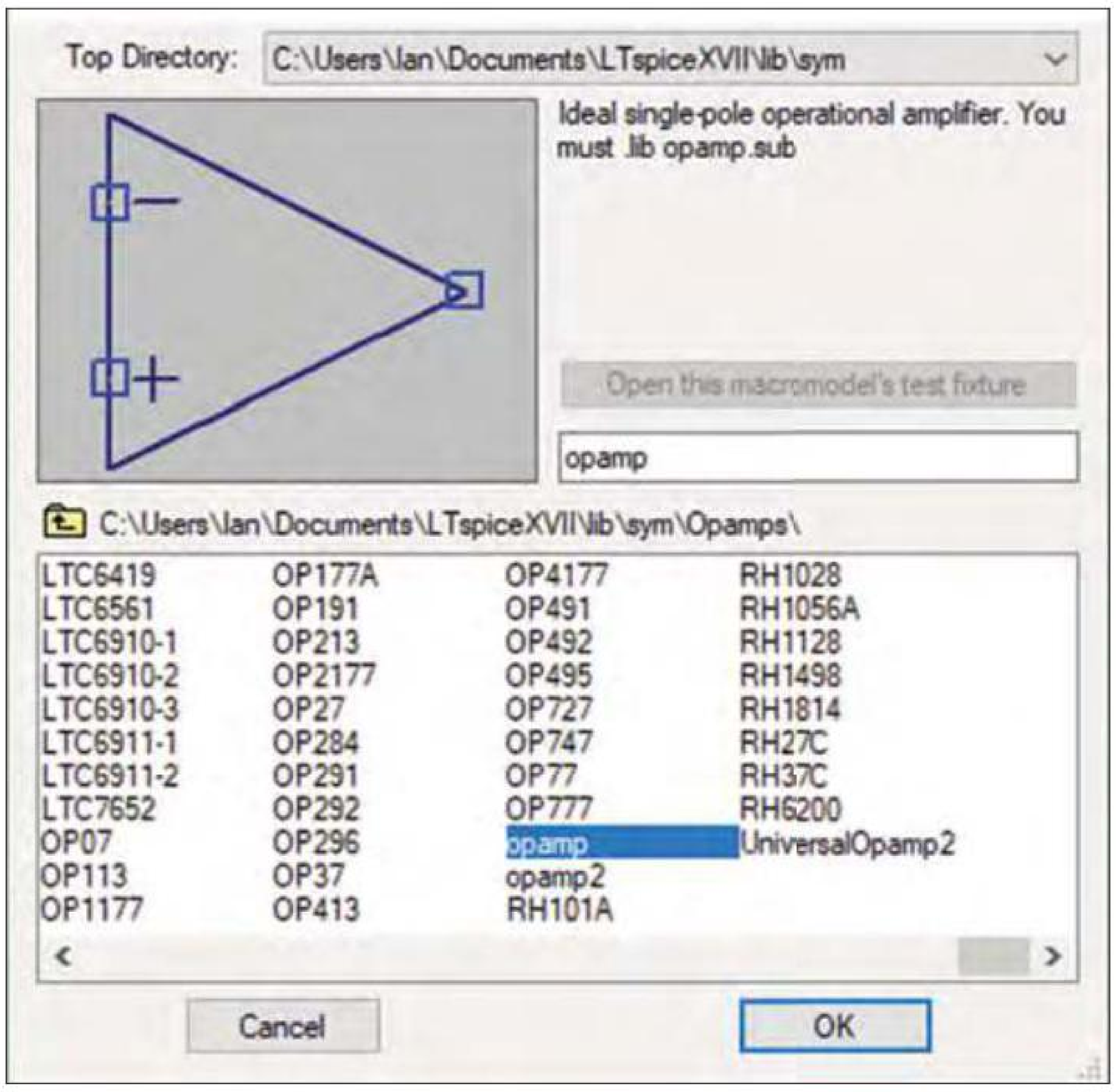
```
.subckt rc in Common out
C1 out Common 10n
R1 out in 1k
.ends rc
```

Referring to the code above, note that between these two directives is a normal SPICE netlist defining the structure of the subcircuit. The `.subckt` directive is followed by the name of the subcircuit (`rc` in this case) and then the pins/ports of the subcircuit. The order of these is as defined using the Pin Properties dialog described above.

The netlist for the top level circuit in Fig.1 contains the following instances of the RC subcircuit, as shown in the code below.

```
XX1 input 0 N001 rc
XX2 N001 0 N002 rc
XX3 N002 0 Output rc
```

Each subcircuit has the name from the schematic (eg, `X1`) prefixed by an `X` (hence `XX1`) because all SPICE subcircuit elements have `X` as an initial letter. We could change



Top Directory: C:\Users\Ian\Documents\LTspiceXVII\lib\sym

Ideal single-pole operational amplifier. You must .lib opamp.sub

opamp

C:\Users\Ian\Documents\LTspiceXVII\lib\sym\Opamps\

LTC6419
LTC6561
LTC6910-1
LTC6910-2
LTC6910-3
LTC6911-1
LTC6911-2
LTC7652
OP07
OP113
OP1177
OP177A
OP191
OP213
OP2177
OP27
OP284
OP291
OP292
OP296
OP37
OP413
OP4177
OP491
OP492
OP495
OP727
OP747
OP77
OP777
opamp
opamp2
RH101A
RH1028
RH1056A
RH1128
RH1498
RH1814
RH27C
RH37C
RH6200
UniversalOpamp2

Fig.7. The location of symbol files is displayed when a component is added to a schematic.

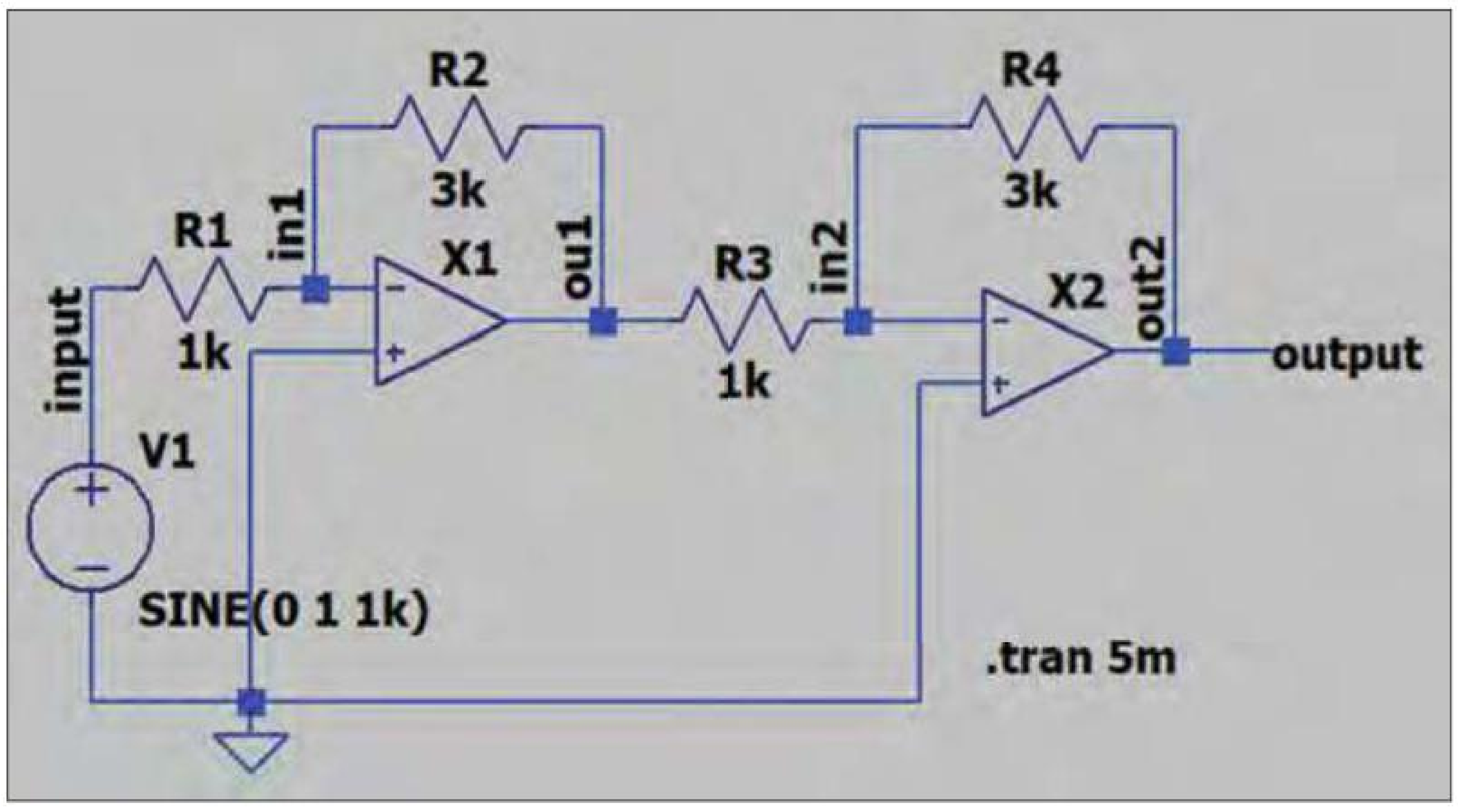


Fig.10. A circuit using the subcircuit in Fig.2 via the symbol in Fig.8.

the name of the RC blocks on the schematic to something else (eg, `RC1`, `RC2`…) and they would still have the correct format in the netlist (eg, `XRC1`, `XRC2`).

Using subcircuits

There are two ways in which a subcircuit can be used – in a one-off hierarchical schematic, or in a library. For a hierarchical schematic the symbols and subcircuit files are all in the same folder as the top-level schematic. The symbol file names match the corresponding subcircuit schematic files names, allowing LTspice to locate the correct files. The user can simply draw symbols and schematics and does not have to deal directly with the subcircuit netlist code.

Placing subcircuits in libraries is more complex. The circuits are stored in the library as netlist text files, not schematics. The library is located in a different part of the file system from the schematic in which it is used, and the symbols and subcircuit definitions are typically in different folders within the library structure. The symbol file needs to contain information on where to find the subcircuit definitions – this is achieved by associating attributes with the symbol using the Attribute Editor (`Edit > Attributes > Edit Attributes` from the menu). Attributes can be used for other information, such as providing descriptive text about the subcircuit, or defining parameter values which can be passed to the subcircuit.

Specific attributes must be set in a particular way for a subcircuit in a library. One approach is to set the Prefix attribute to `X` (the initial letter of subcircuit elements), the SpiceModel attribute to the name of the model (eg, `rc` in the example above) and the ModelFile attribute to the full file path of the file containing the subcircuit netlist. There are variations on how attributes can be set up for symbols in a library – refer to the help on attribute editing for details. Symbols for use with hierarchical schematic have all their attributes blank.

Walkthrough

This walkthough provides step-by-step instructions for creating a hierarchical schematic, assuming familiarity with aspects of LTspice covered in the earlier articles. This is followed by instructions for adding a new component to the library.

Create a new folder for your work. Open a new schematic in LTspice and draw the circuit shown in Fig.2, saving it as `AmpModel.asc`. Copy (don't move) the symbol file `opamp.asy` from the LTspice library to the same folder as the schematic and rename it `AmpModel.asy`. To find the location of the file, use the Component function as if you were going to add the op amp to the schematic, and check the location displayed in the dialog (see Fig.7). Cancel the component add.

In LTspice open the local copy of `AmpModel.asy` symbol file (see Fig.8) – you will need to change the file type dropdown to Symbols (`*.asy`) in the file browser. Right click on both input pins in turn and use the Pin/Port dialog (like Fig.5) to change the pin names to those in Fig.2 (`invin` to `inn`, and `noninvin` to `inp`). The output has the same name (out) so does not need changing. While doing this, note the netlist order for each pin (`inn 1`, `inp 2`, `out 3`). This symbol was originally part of a library, which means that it is not configured correctly for use in a hierarchical schematic. To rectify this do `Edit > Attributes > Edit Attributes`. This will open the Symbol Attribute Editor. Change the Symbol Type: dropdown from Cell to Block and remove all the text values from the value column. To edit the text, double click on it first. When you have finished, the dialog should look like Fig.9. Click OK. Save the symbol.

Create a new schematic as shown in Fig.10 and save it in as `TwoStageAmp.asc`. To place the amplifiers use the Component button (or `Edit > Component`) and change the Top Directory dropdown from the default (the install location of the LTspice libraries) to the folder with the

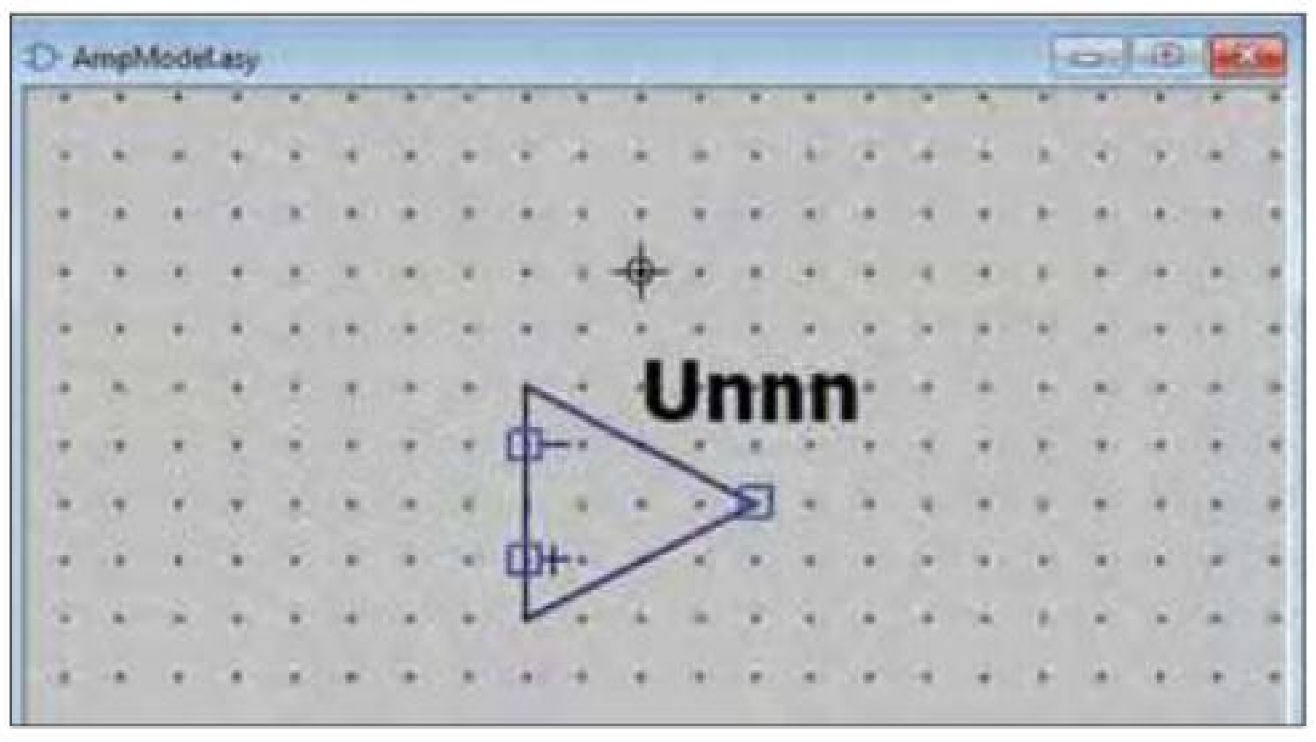
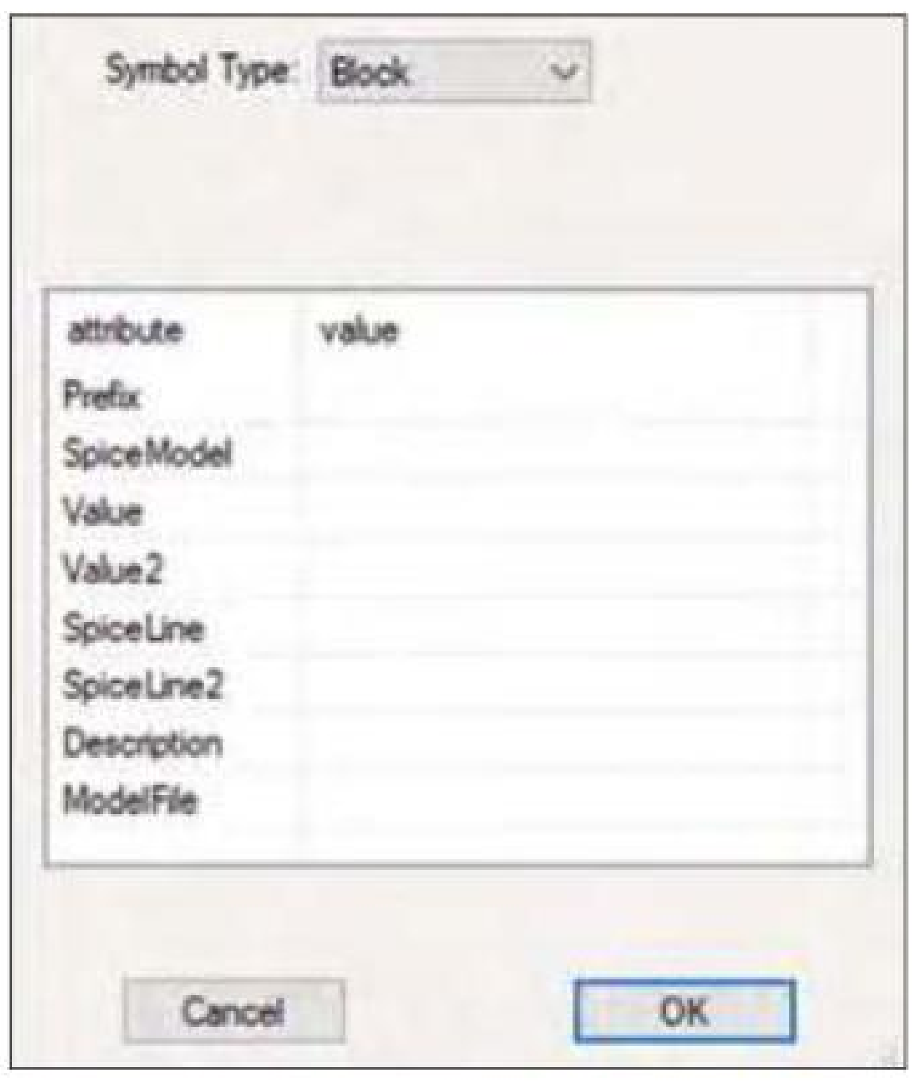


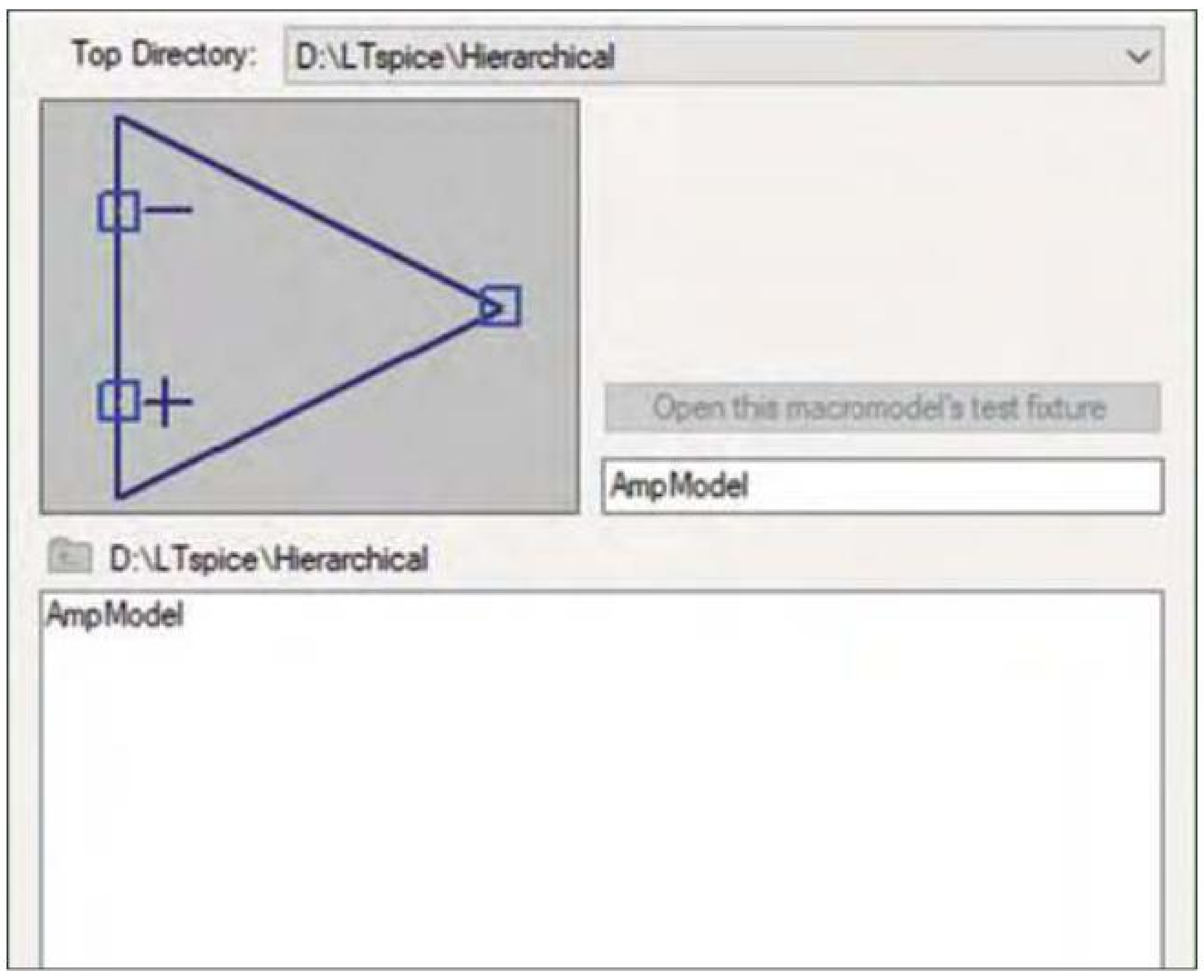
Fig.8. Copied op amp symbol in editor saved as `AmpModel.asy`.



Symbol Type: Block

attribute	value
Prefix	
SpiceModel	
Value	
Value2	
SpiceLine	
SpiceLine2	
Description	
ModelFile	

Fig.9. Symbol Attribute Editor – for hierarchical schematics all attributes should be left blank and the symbol type should be 'Block'.



Top Directory: D:\LTspice\Hierarchical

AmpModel

D:\LTspice\Hierarchical

AmpModel

Fig.11. Adding a subcircuit to a hierarchical schematic

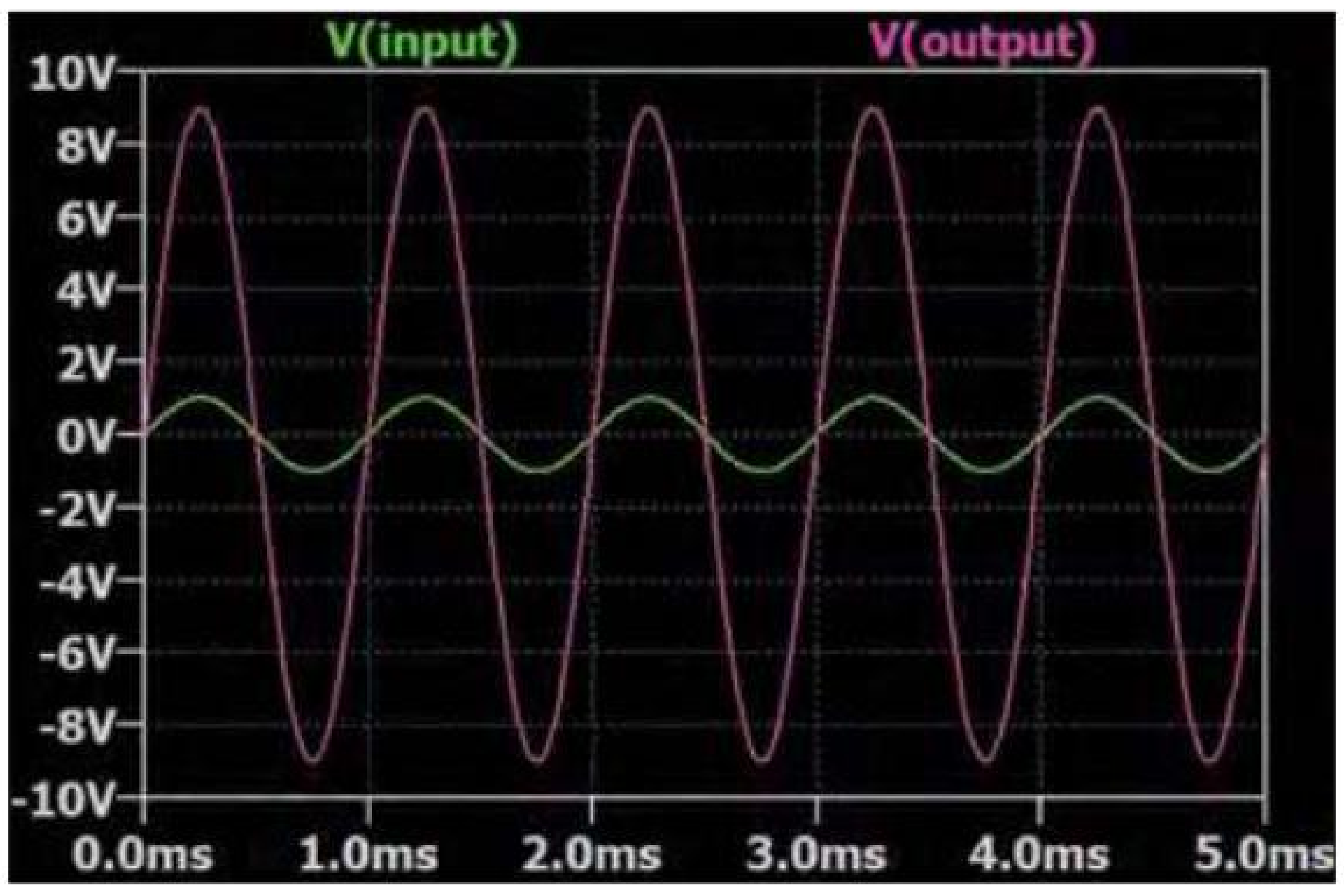


Fig.12. Results from simulation of the Fig.10 circuit.

current design in it (`D:\LTspice\Hierarchical` in this example) – see Fig.11. This will make symbols in the current folder available to place on the schematic. Place two AmpModel amplifiers on the schematic (as in Fig.10) and note that they are named `X1` and `X2` on the schematic.

Set up the `V1` source as a 1.0V 1.0kHz sinewave and configure a transient simulation running for 5ms, as shown in Fig.10. Save the schematic. Running the simulation should produce the results shown in Fig.12 – the output amplitude is 9V – nine times the input voltage, as would be expected from two inverting amplifiers in series, each with a gain of –3.

For the next part of the walkthrough we are going to create a copy of the amplifier model in the LTspice library. We can create a new folder that will appear on the list of components when editing schematics. Any symbol files placed in here will be available to add to the schematic. Find the location of the LTspice symbols library (`sym` folder) on your computer (eg, `C:\Users\username\Documents\LTspiceXVII\lib\sym`) and create a new subfolder in this with a name of your choice (eg, `MyComponents`). Copy the `AmpModel.asy` symbol file to this folder.

Also create another new folder for the subcircuit netlist – a subfolder of `…\Documents\LTspiceXVII` may be appropriate. We need to put the netlist text for the subcircuit in a `.lib` file in this folder. To do this, with the TwoStageAmp schematic selected in LTspice do `View > SPICE Netlist`, and, highlight the text of just the subcircuit (as follows), then use `CTRL-C` to copy it.

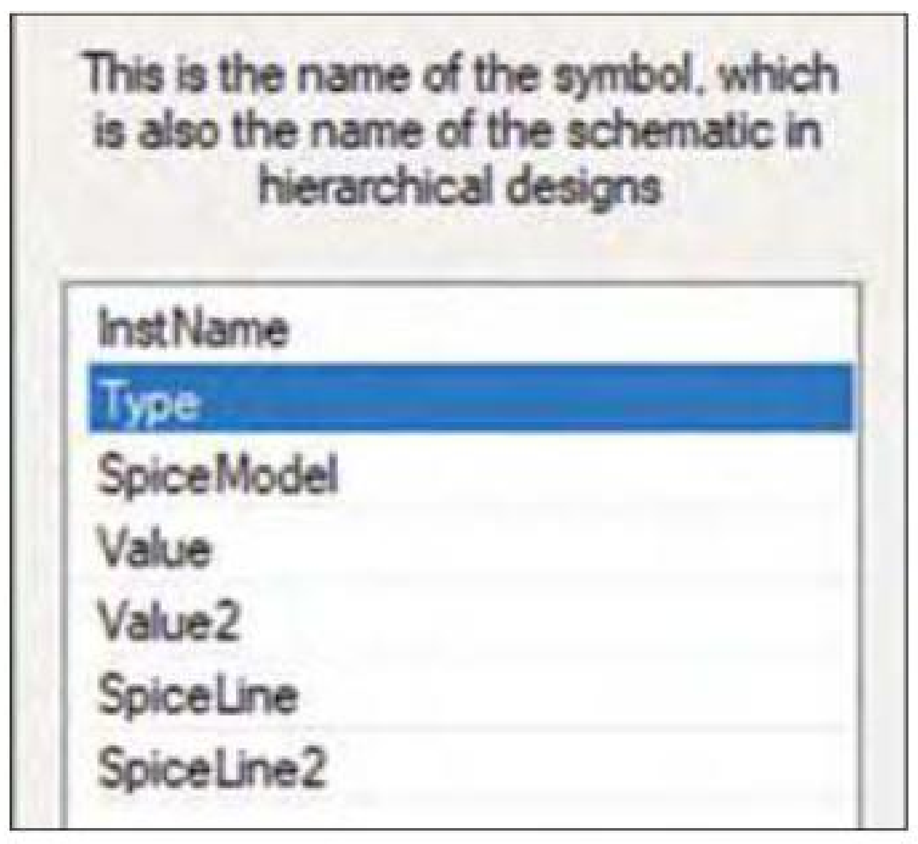
```
.subckt ampmodel inn inp out
Rout N001 out 10
Rind inn inp 50Meg
E1 N001 0 inp inn 100k
Rincn inn 0 10Meg
Rincp inp 0 10Meg
.ends ampmodel
```

Open a text editor (Notepad will do) and paste the above code into it. Then save this to a file called `AmpModel.lib` in the folder you created for your library component netlists (eg, `…\Documents\LTspiceXVII\MyComponents`).

In LTspice, open the `AmpModel.asy` symbol from the library location you just created and do `Edit > Attributes > Edit Attributes`. In the Symbol Attribute Editor (see Fig.13) set Symbol Type to Cell, and set the attribute values as follows: Prefix to X, SpiceModel to AmpModel, and ModelFile to the full path to the `.lib` file that you just created. You can also add a description of the component if you want. Click OK. To add the device name to the symbol do `Edit > Attribute > Attribute Window`. Click on Type (see Fig.14) and then OK. Click on the symbol to place the text `<Type>` where you want it. Save the symbol.

Save the TwoStageAmp schematic if you have not done so already. Then delete the two amplifier symbols and `Save as` to save the file with a different name, ideally in a new folder. Close LTspice and run it again – this is so that it rereads the library contents. Open the new copy of the two-stage amplifier schematic and click the component button. Note that the folder you created [`MyComponents`] now appears on the list. Double-click on it and note that AmpModel (the symbol file you placed in `MyComponents`) is listed. Click on it – you will see its symbol and any description text that you entered (see Fig.15). Add two instances of the amplifier to the schematic in the gaps – note the name `ampmodel` is displayed where you placed `<Type>` on the symbol. Save the schematic and run the simulation to confirm correct operation.

Fig.14. Dialog for selecting attributes to show on the symbol.


This is the name of the symbol, which is also the name of the schematic in hierarchical designs

InstName
Type
SpiceModel
Value
Value2
SpiceLine
SpiceLine2

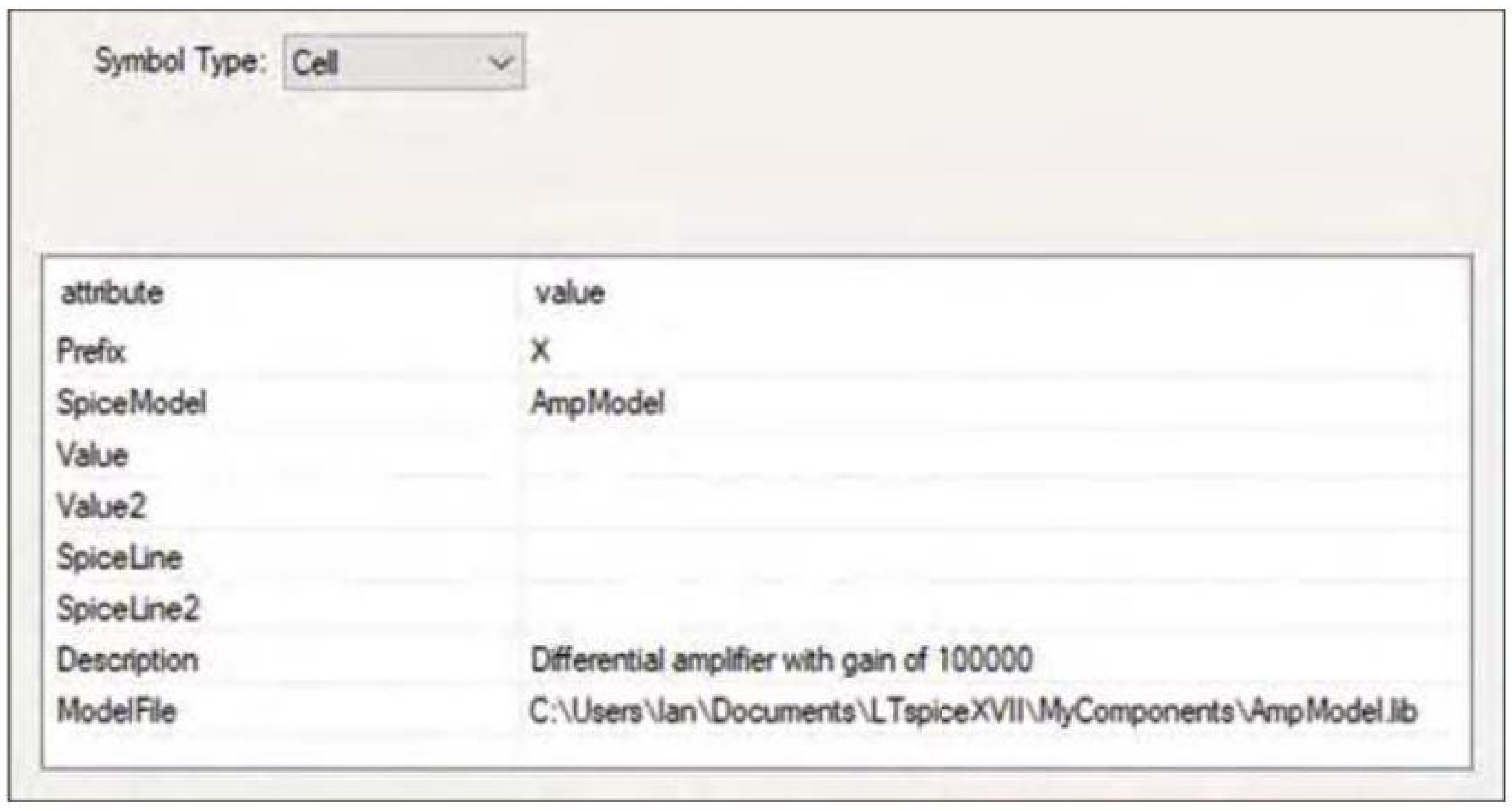
Downloading models for LTspice

The process of handling downloaded models is similar to the second half of the walkthrough above. When you download a SPICE model you typically get a subcircuit netlist, which should be put in a `.lib` file in a suitable location (change the extension if required). You then need to create or copy and edit a symbol for the component and put this in a subfolder of the LTspice `sym` folder. Edit the symbol to set the pin names and netlist order – you will need to look at the `.subckt` statement in the downloaded model to find this information. For copied symbols, pin names only matter if they are displayed on the symbol. Edit the attributes to set Type (to Cell), Prefix (to `X`) SpiceModel (to the name found in the `.subckt` statement) and ModelFile (to the downloaded `.lib` file path).

As an example, if you search the internet for '741 SPICE model' you will find a link on the Texas Instruments site for a 'LM741 PSPICE Model' (note Pspice, not LTspice). The download provides an `LM741.MOD` file, which can be renamed to `LM741.lib`. In this file you will find:

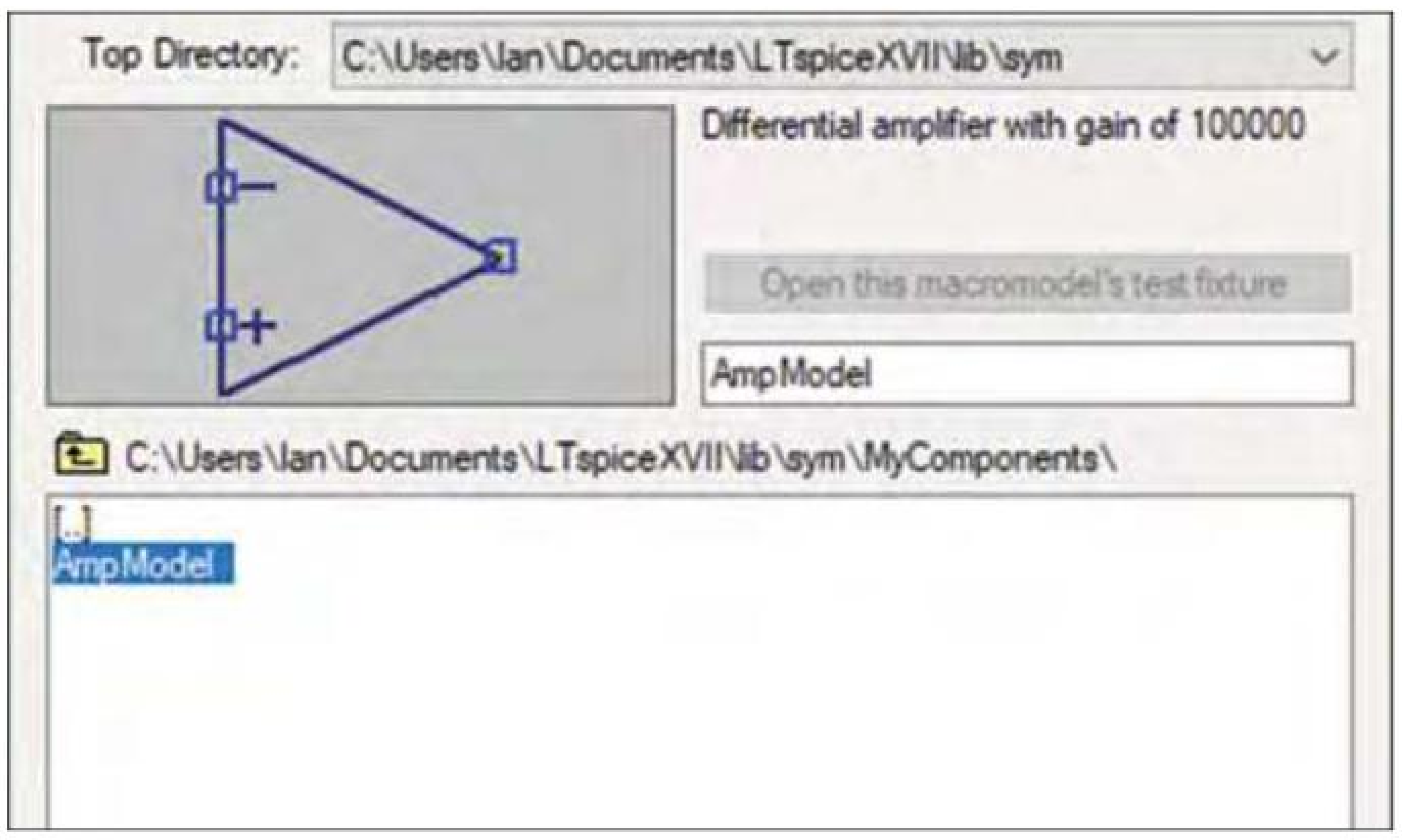
```
.SUBCKT LM741/NS 1 2 99 50 28
```

The pins have numerical names, but comments in the file identify their function. The existing opamp2 symbol could be copied and edited to create the symbol – the pin order is the same, which helps. The `/NS` in the `.subckt` statement is understood by Pspice, but not LTspice and needs to be deleted (and the file saved) before it can be used. SPICE is a de facto standard, but not all versions are exactly the same and it is sometimes necessary to make edits such as this to downloaded models.


Symbol Type: Cell

attribute	value
Prefix	X
SpiceModel	AmpModel
Value	
Value2	
SpiceLine	
SpiceLine2	
Description	Differential amplifier with gain of 100000
ModelFile	C:\Users\Ian\Documents\LTspiceXVII\MyComponents\AmpModel.lib

Fig.13. Symbol Attribute Editor for a component in the library.


Top Directory: C:\Users\Ian\Documents\LTspiceXVII\lib\sym

Differential amplifier with gain of 100000

AmpModel

C:\Users\Ian\Documents\LTspiceXVII\lib\sym\MyComponents\

[..]
AmpModel

Fig.15. AmpModel is available from the library when adding components.

PIC n' Mix

Mike Hibbett's column for PIC project enlightenment and related topics

Small, cheap and powerful – Part 2

In last month's article we introduced the PIC10F family of microcontrollers, the tiniest and most feature-limited processor series available from Microchip Semiconductors. It's also a very cheap processor, available for less than £0.50 in singles.

It's not the cheapest microcontroller on the market, even from Microchip. For Microchip, that prize goes to the ATTiny processor. The PIC10F, however, is based on the PIC architecture that we are familiar with, and all of the usual Microchip tools (MPLAB X IDE, PICkit 3 for example) work well with it. The format of the datasheet will be familiar to anyone who has used other PIC microcontrollers, so getting productive with the device will be easy if you have played with other PIC microcontrollers. It's a very-feature-limited part, so it will also be easy to understand for anyone unfamiliar with microcontrollers, but who wants to get to know what goes on 'under the hood'. In short, it's an ideal learning platform.

These microcontrollers may appear too limited to be practical in a 'professional' manner, but they do find their niches. A PCB design produced by the author for an industrial application, shown in Fig.1, has been used in over 1500 installations. And this was the smallest of the PIC10F family, the PIC10F200. Only 35% of the available code space was used, so there is a good chance we can have some fun with the PIC10F202, which packs twice the available memory – a whopping 512 bytes!

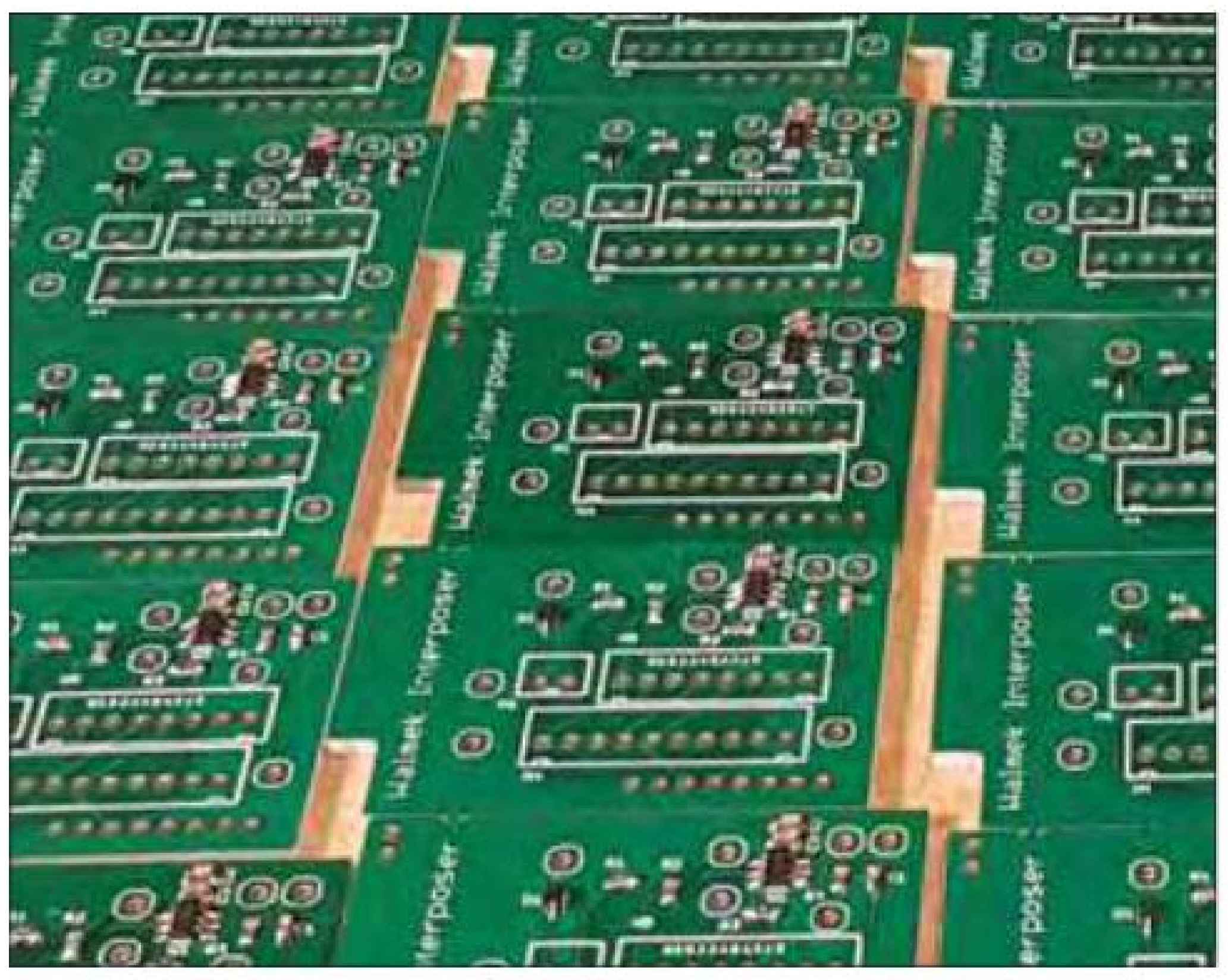


Fig.1. A commercial application of the PIC10F200

This month's project

Let's take a look at this month's project, a smart power switch. Here we are controlling a relay, which can be used to control an external circuit with any amount of power requirements, depending on the type of relay used. To simplify the design we have shown a non-latching relay, the cheapest and most readily available form. These relays take several tens of milliamps while energised, but this circuit can be improved with the addition of a single transistor and two resistors to make use of a latching relay, which would mean the energy consumption is significantly lower when the attached circuit is powered.

A momentary push switch is connected to GPIO pin GP1, and we make use of an internal pull-up resistor inside the PIC to 'pull' the input pin to the positive supply voltage. The push button shorts the input pin to ground, which we will use to wake the processor from sleep, and then run software to perform whatever functionality we decide on. The internal pull-up resistor is a very high value, so there is no problem with pulling the input pin directly to ground. This is exactly the functionality the chip designers had in mind when they provided internal pull-up resistors.

If you have read last month's article you may have noticed that part of the circuit has changed slightly. Last month, we showed the push switch going to GP2. For some odd reason only GP0, GP1 and GP3 have internal pull-up resistors and the 'wake from sleep' functionality; GP2 is just a normal input/output pin. These kind of 'gotcha!' issues occur frequently in microcontroller use and this is an example of why carefully reading the datasheet is important. The final circuit is in Fig.2.

PIC n' Mix files

The programming files discussed in this article are available for download from the *PE* website.

The GPIO pin GP2 is used to drive an LED as a simple user interface (it really doesn't get simpler than this) and we can turn the LED on, off or flash at different rates to provide feedback. Pin GP0 is used to provide the actual power on and off control, with a FET being configured as a 'low-side' switch to an arbitrary load – a relay in our case, but it can be pretty much anything. For testing, we will replace the relay and its protection diode with a simple LED and series resistor for visual feedback.

If the power requirements are low enough it is possible to replace the pricier FET with a simple cheap NPN transistor. If using a FET, be careful to pick one with a low gate threshold. Even this FET, an IRL510, will not be fully turned on with this design and the maximum current that it can pass from drain to source will probably be limited to 1A; but that's more than enough for controlling a relay. We chose the

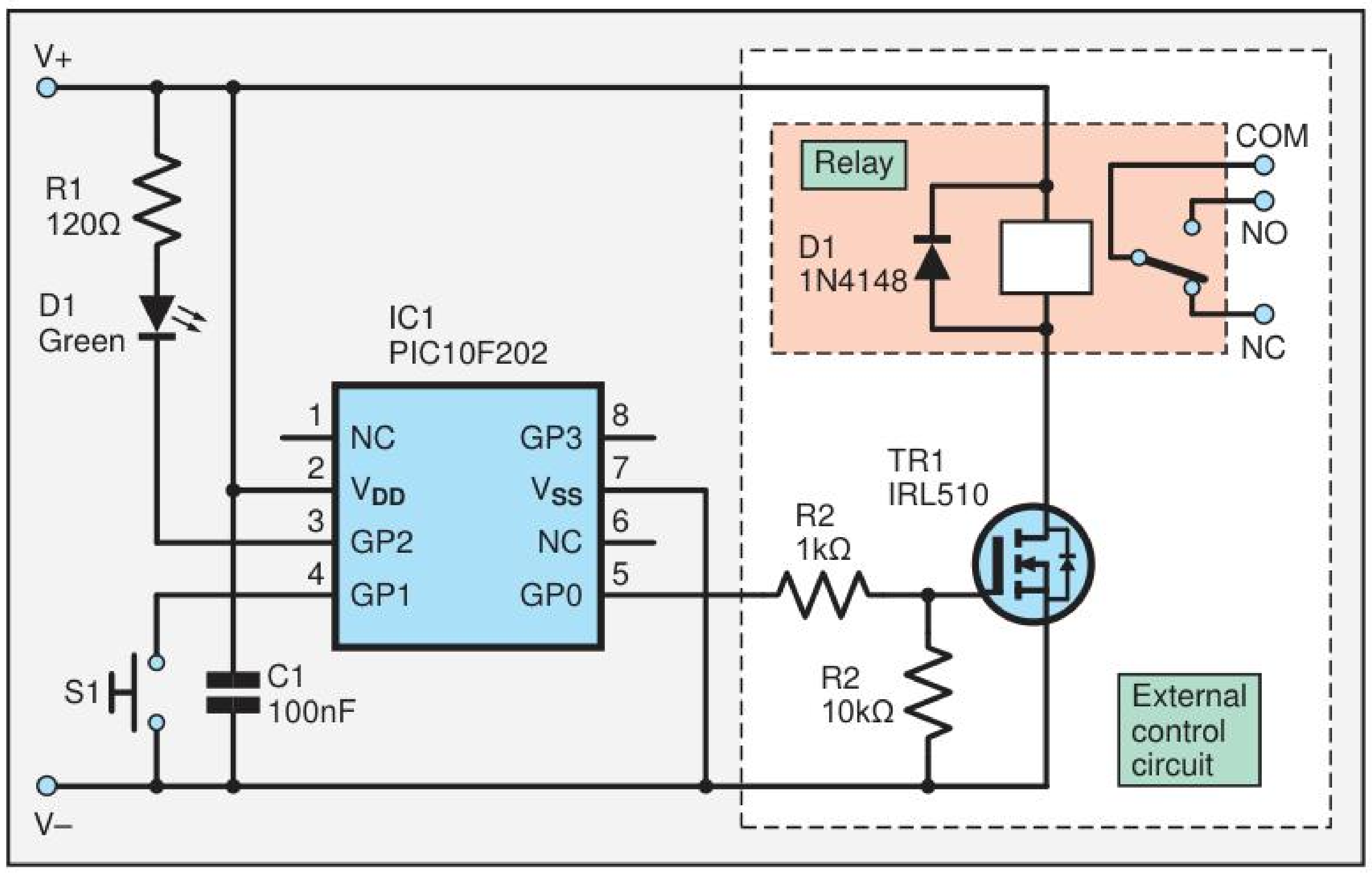


Fig.2. This month's project schematic - a smart switch

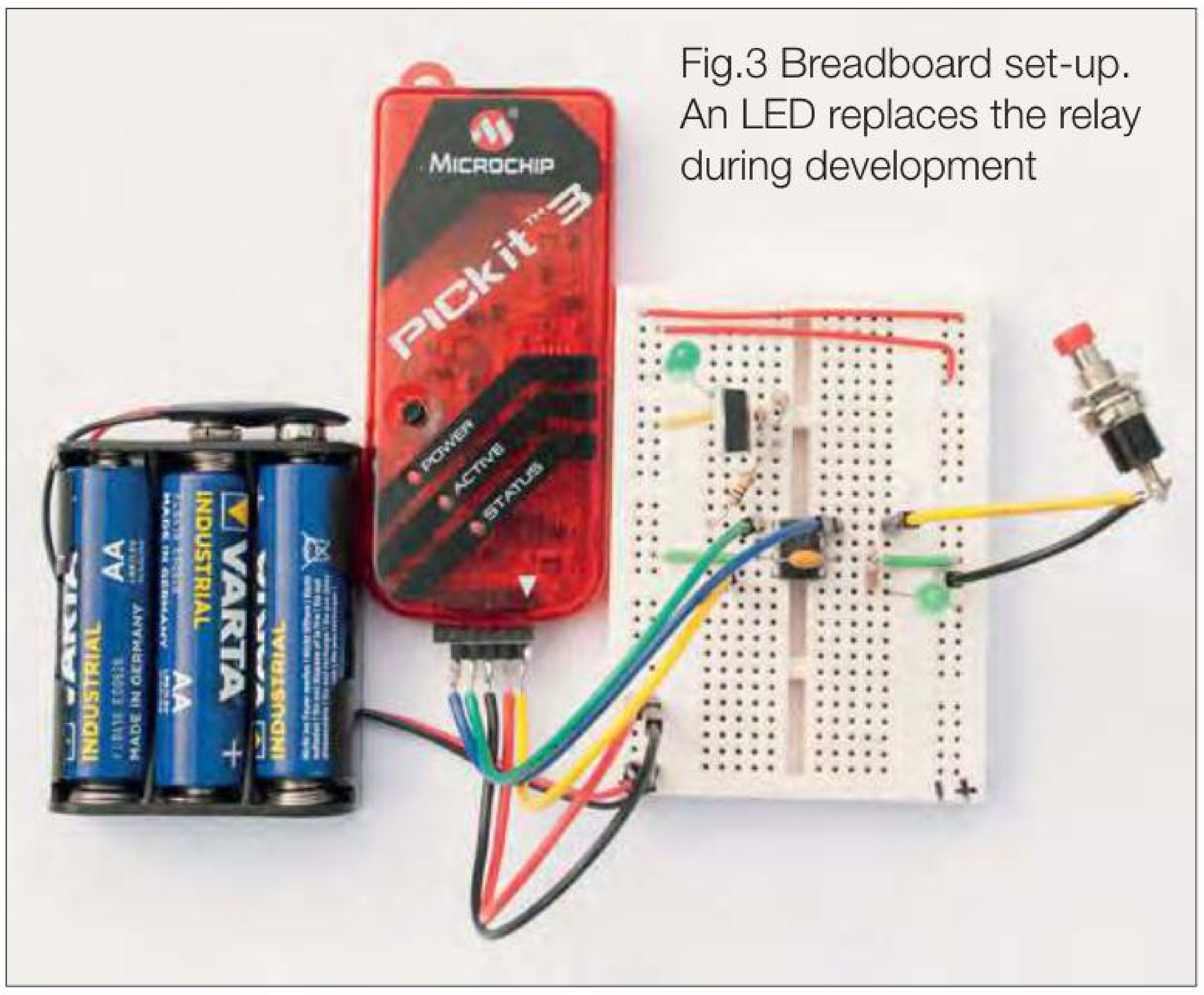


Fig.3 Breadboard set-up. An LED replaces the relay during development

IRL510 simply because we had some devices to hand in the lab.

Power supply considerations

As mentioned earlier, there are latching relays available on the market, and readily available to hobbyists at low cost. These devices have two coils and require two GPIO pins and two transistors to drive them. Although that increases the complexity of a circuit a little, latching relays provide the benefit of requiring just a short pulse of current to change from open to closed, and a short pulse to change them back. Using a latching relay means this circuit would be able to operate for around 8 to 10 years on four AA batteries, for example. The Panasonic relay TX2-LT-4.5V-TH is an example of a low-cost mains switching latching relay, if you are following along at home, *and* have the knowledge and experience to build mains voltage circuits safely.

The processor can run from a voltage range of 2V to 5.5V, but has a 'maximum' voltage of 6.5V, so we can run this circuit from two, three or four alkaline batteries. As a compromise between physical size and operational life of the circuit we have chosen to go with three AA batteries, giving an expected voltage range of 2.7V to 4.5V. We can evaluate the performance of 4.5V relay switching capabilities at 2.7V later on, and take some measurements to give a more accurate estimate of how long the batteries can be expected to last.

In this design, which avoids the use of a regulator to provide a constant voltage to the circuit thanks to the processor's ability to operate with a wide range of voltages, we had to take some care in the choice of the series resistor to the LED. Taking careful measurements with a variable power supply and a multimeter, a 5mm green LED draws 20mA at 2.3V, but at 2.1 it draws 10mA and is still perfectly visible. It's clearly visible down to 1.9V, and draws around 5mA. With that in mind, we choose a resistor to provide 20mA when the batteries are at full voltage, 4.5V. Ohm's law gives 110Ω; so let's try 120. Testing on a variable power supply showed that the LED is perfectly visible across the fully battery range of 2.7V to 4.5V – 120Ω it is then.

So what exactly will we get this circuit to do? Let's start with some basic functions and improve iteratively until we run out of memory space. In order of preference, let's try to drive this circuit to do:

1. De-bounce the push switch, turn output on if still active after de-bounce.
2. Turn the output off after a programmed delay (say, 30 minutes, or longer).
3. If the output is on and the button is pressed again for a long period (say, 3s) turn the output off immediately.
4. If the output is on and the button is pressed, flash the LED to indicate power off is about to happen, if the button is continued pressed.

First, we build the test circuit up on breadboard, as shown in Fig.3. We've added an LED and series resistor in place of the relay so we can see what is going on. A 0.1-inch header strip is used to provide the programming interface signals to a PICkit 3 with jumper headers onto the breadboard. A PICkit 4 will work just as well, if you have one.

To test everything works, let's create an 'empty' project within MPLAB IDE, and download it to the board.

Creating an initial project

First, ensure that you have the XC8 compiler installed and the software for your programming hardware (PICkit 3 or PICkit 4) already installed and plugged in.

Now let's go ahead and write some software. Open up MPLAB IDE and select the 'File->Close All Projects' menu options to give us a clean workspace.

Select 'File->New Project', and in the dialogue that appears, click on 'Standalone Project' followed by 'Next'. In the next dialogue, select PIC10F202 from the 'Device' drop down menu, and click 'Next'. Click 'Next' again. Now, click on your PICkit device in the list – the serial number of the device should be shown – and click 'Next'. Now click on the XC8 complier entry in the list of complier tool-chains. In the next dialogue type in a name for the project – 'test1', for example – and click 'Finish'.

This creates a project for you within MPLAB; the files that define which processor you are using and what complier tools you are using – but no source files. So let's go ahead and create our single program source file, which we will call **main.c**. To do this, right-click over the text 'Source Files' in the list of project items, select 'New' and then **main.c**. A dialogue will appear; type in the name of our file, 'main', and click 'Finish'. This will create a file called **main.c** in the project directory.

You will now be presented with a series of windows within the IDE similar to Fig.4. Our new source file has been created for us and populated with a very basic 'C' program – nothing more than the 'return' statement inside a typical 'main' function. Clearly, this program is going to do nothing interesting. Yet it is a valid program; click on the hammer icon on

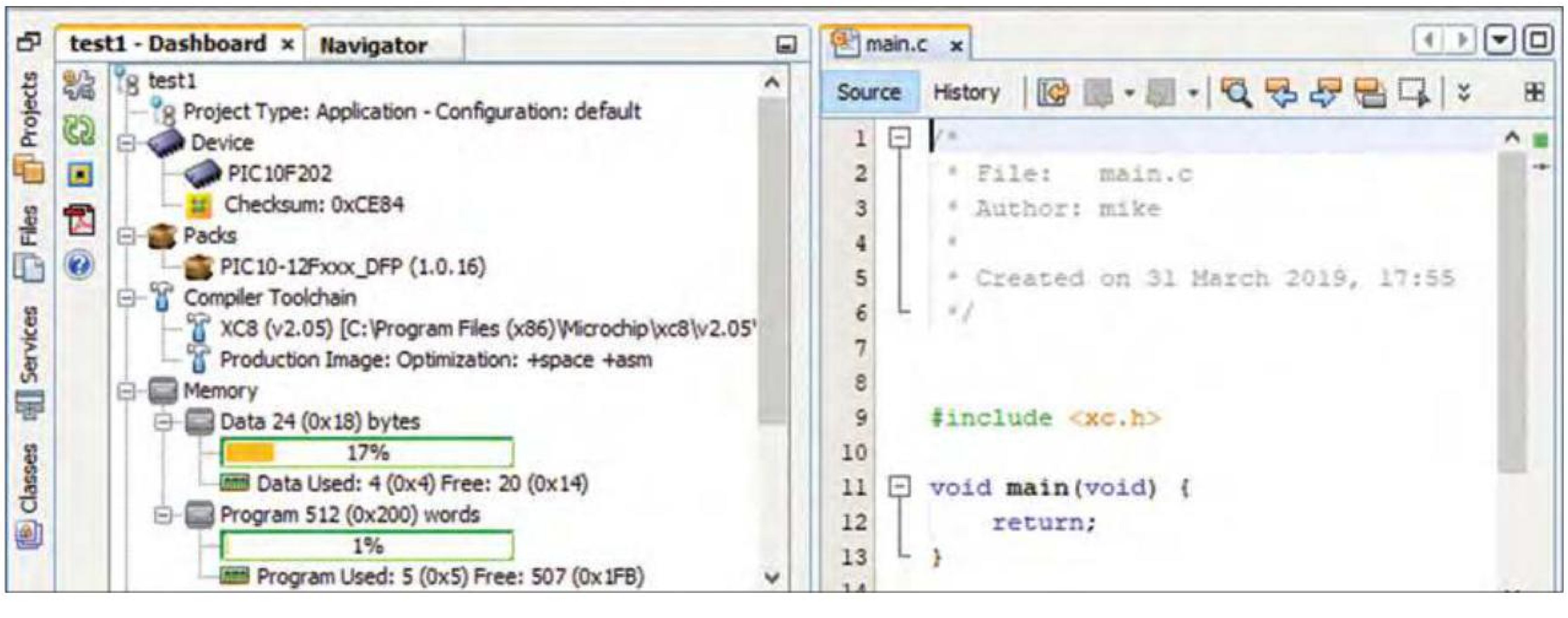


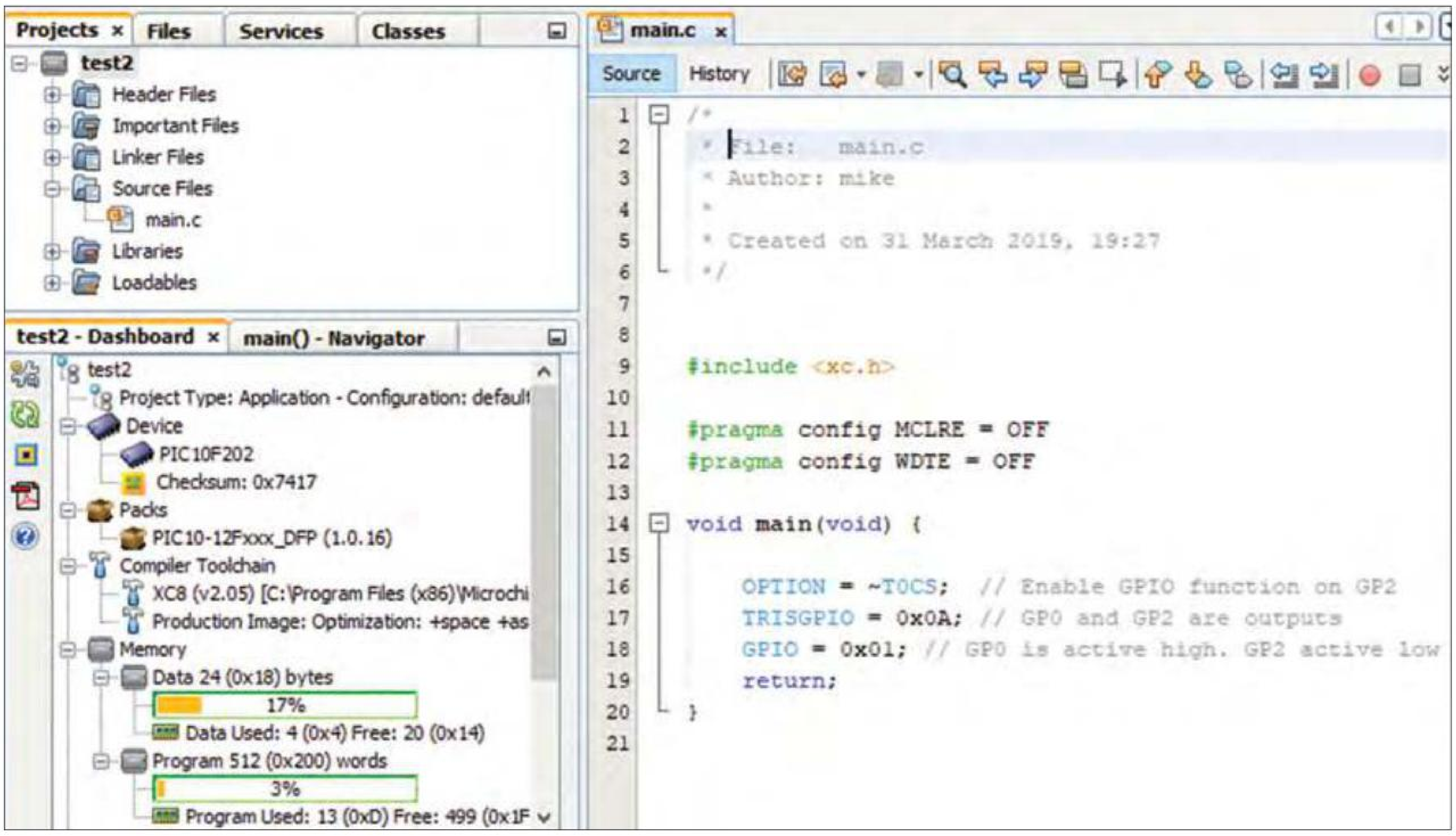
Fig.4. Our first C program, which does nothing!

test1
Project Type: Application - Configuration: default
Device
PIC10F202
Checksum: 0xCE84
Packs
PIC10-12Fxxx_DFP (1.0.16)
Compiler Toolchain
XC8 (v2.05) [C:\Program Files (x86)\Microchip\xc8\v2.05'
Production Image: Optimization: +space +asm
Memory
Data 24 (0x18) bytes
17%
Data Used: 4 (0x4) Free: 20 (0x14)
Program 512 (0x200) words
1%
Program Used: 5 (0x5) Free: 507 (0x1FB)

```
/*
 * File:   main.c
 * Author: mike
 *
 * Created on 31 March 2019, 17:55
 */


#include <xc.h>

void main(void) {
    return;
}
```



test2
Project Type: Application - Configuration: defaul
Device
PIC10F202
Checksum: 0x7417
Packs
PIC10-12Fxxx_DFP (1.0.16)
Compiler Toolchain
XC8 (v2.05) [C:\Program Files (x86)\Microchi
Production Image: Optimization: +space +as
Memory
Data 24 (0x18) bytes
17%
Data Used: 4 (0x4) Free: 20 (0x14)
Program 512 (0x200) words
3%
Program Used: 13 (0xD) Free: 499 (0x1F

```
/*
 * File:   main.c
 * Author: mike
 *
 * Created on 31 March 2019, 19:27
 */


#include <xc.h>

#pragma config MCLRE = OFF
#pragma config WDTE = OFF

void main(void) {

    OPTION = ~TOCS;  // Enable GPIO function on GP2
    TRISGPIO = 0x0A; // GP0 and GP2 are outputs
    GPIO = 0x01; // GP0 is active high. GP2 active low
    return;
}
```

Fig.5. The simplest program: light two LEDs

the top horizontal menu bar to build the project, and you should be greeted with a successful build.

The window in the bottom left part of the IDE shows the amount of data and program memory used by our application, once we download it to the chip. Five words of Flash memory, but four of our valuable data bytes – that's 17% gone and we haven't even started! What's going on?

This memory utilisation is what the 'C' language itself requires before your code – even everything inside your 'main' function gets invoked. This is referred to as the 'start-up code', or more technically `cstartup` or `crt0`, depending on what kind of system the program is running on. On larger systems with an operating system the start-up code can run into hundreds of bytes, if not kilobytes. The 'C' compiler for the Microchip processor family has been carefully optimised to minimise the overhead where possible, and these handful of memory locations can be considered the 'tax' for using a high level language. It's a very small tax indeed – thank you Microchip!

Before we move on to writing actual code, there is one change we have to make to the project configuration files. The internal oscillator of each processor is measured during manufacture, and a calibration offset value stored in one of the Flash memory locations. On processor start-up, if that value is loaded into a calibration register in the processor, the device's oscillator will run with an accuracy of 1%. It's a feature that is well worth the two-word 'tax' on Flash memory required to apply this calibration correction value. By enabling the feature, the code to do this is inserted into you file during start-up, so you do not need to write any code to handle it. So let's go ahead and do that.

In the top left-hand section of the IDE, click on the 'Projects' tab, then right-click over the project's name (test1), and select 'Properties' from the long list of options that appear (it's the last in the list.) In the dialogue that appears after this, click on the 'XC8 Linker' item in the 'Categories' list, then click the check box for 'Calibrate oscillator', followed by 'OK'. If you now re-build the project you will see that the program memory use has gone up by two more bytes. Now, we are ready to write some code.

Baby Steps

Let's start by writing a trivial program, one that just turns on the two LEDs. You can see the lines we have added to **main.c** in Fig.5. If you enter these five lines and click the 'Build' icon again, you will see the data usage remain unchanged, while the program memory utilisation has gone up by six words. This is good; the assembly language equivalent is six words too. You could not have written this more efficiently in assembly language.

You may be wondering what the two `#pragam` statements are at the beginning of our code. These are instructions to the MPLAB IDE software to write some special values to the configuration register, CONFIG, into the processor during device programming. The CONFIG register is a special Flash memory register, which is outside of the normal 'code' memory address space. This register, a single byte with just three settings, is used to define parameters of the chip that must be configured before your application starts running. In our case, we are telling the processor to not enable the watchdog timer – we will talk about that more later – and also to configure the MCLRE pin as a GPIO pin.

The first line of code that we added adds another early start-up requirement – configure the GP2 pin as a GPIO pin rather than it's default setting as a timer input pin. That one took us by surprise – but is typical of why it's important to read the datasheet carefully. Even on a simple processor.

To test everything is working, go ahead and download the code to your breadboard, making sure you have the 4.5V battery supply connected. The download icon is to the right of the build icon – a picture of a chip with a down arrow. You will be prompted with a 'CAUTION' dialogue, just click 'OK' to continue.

If all goes well, you should be greeted with the final line of the 'output' window announcing: `Programming/Verify complete`

However, if you received a warning message and the programming fails, then the issue – assuming everything is connected correctly – is that the LED on pin GP0 is loading the programming interface. If that is the case, remove the resistor on pin GP0 temporarily – it can load the programmer pin, interfering with communication. You can replace it after programming is complete. For us this wasn't necessary, but circuit variations may affect your operation, so just take note. At the end, if successful, both LEDs will be illuminated.

We could now go straight into the software design, but let's take some time to look at the building blocks we need to use to construct our project, and what kind of impact they have on the amount of code space they need when developed using the 'C' programming language. We need to be careful about every byte used, and so it will be good to see which code sequences are efficient, and which are not.

Debugging

Before we do that, some bad news about developing software on this processor. The PIC10F is so simple a device Microchip had to leave some features out, and that included simple debugging. To debug, you need to add a special 'debug header', a requirement the datasheet does not mention. The reason for adding this feature is that Microchip recognise that on small pin count parts, you may be forced to use the programming pins

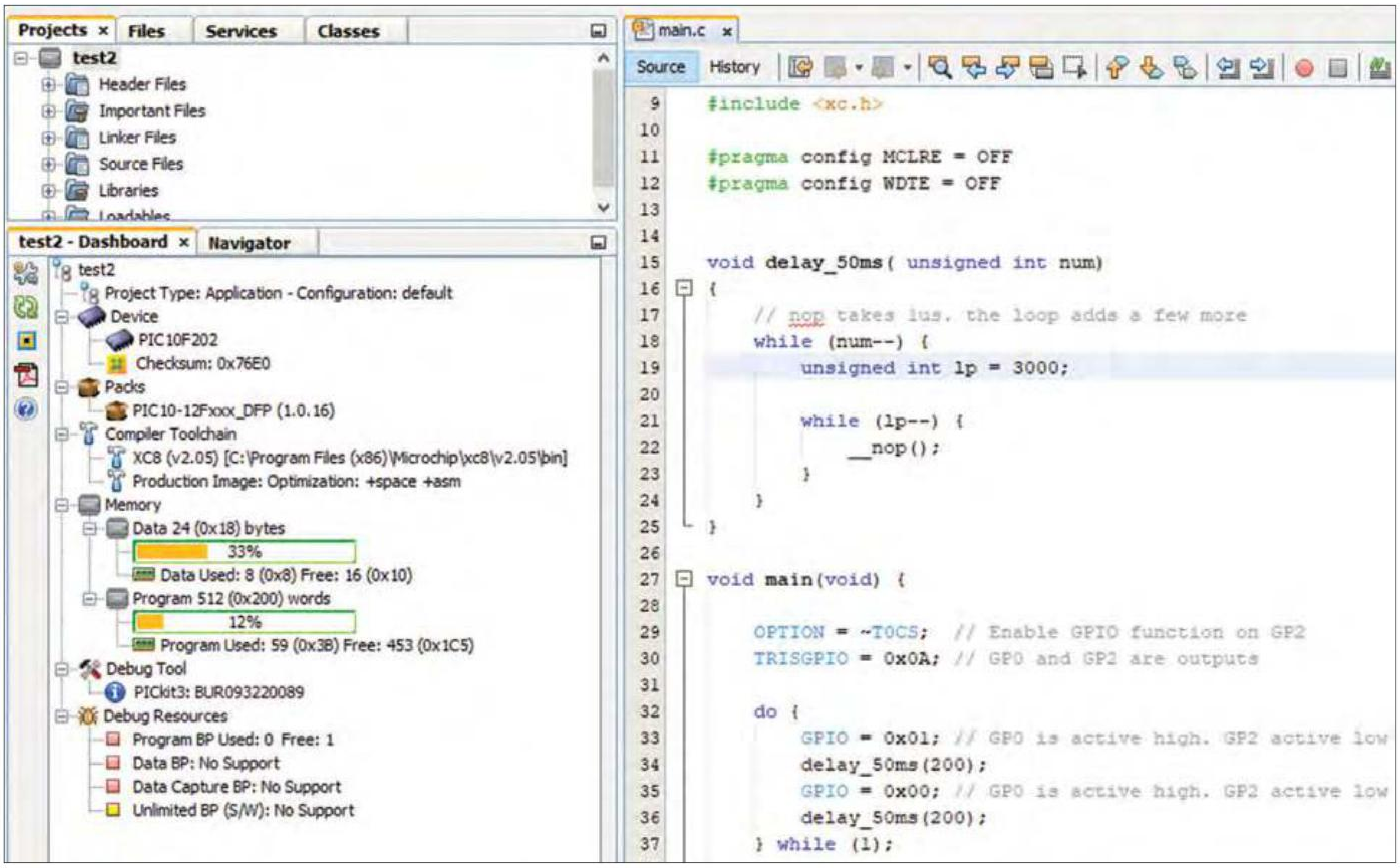


Fig.6. The initial delay function and test code

```
test2
  Header Files
  Important Files
  Linker Files
  Source Files
  Libraries
  Loadables
```

```
test2
  Project Type: Application - Configuration: default
  Device
    PIC10F202
    Checksum: 0x76E0
  Packs
    PIC10-12Fxxx_DFP (1.0.16)
  Compiler Toolchain
    XC8 (v2.05) [C:\Program Files (x86)\Microchip\xc8\v2.05\bin]
    Production Image: Optimization: +space +asm
  Memory
    Data 24 (0x18) bytes
      33%
      Data Used: 8 (0x8) Free: 16 (0x10)
    Program 512 (0x200) words
      12%
      Program Used: 59 (0x3B) Free: 453 (0x1C5)
  Debug Tool
    PICkit3: BUR093220089
  Debug Resources
    Program BP Used: 0  Free: 1
    Data BP: No Support
    Data Capture BP: No Support
    Unlimited BP (S/W): No Support
```

```c
#include <xc.h>

#pragma config MCLRE = OFF
#pragma config WDTE = OFF


void delay_50ms( unsigned int num)
{
    // nop takes 1us. the loop adds a few more
    while (num--) {
        unsigned int lp = 3000;

        while (lp--) {
            __nop();
        }
    }
}

void main(void) {

    OPTION = ~T0CS;  // Enable GPIO function on GP2
    TRISGPIO = 0x0A; // GP0 and GP2 are outputs

    do {
        GPIO = 0x01; // GP0 is active high. GP2 active low
        delay_50ms(200);
        GPIO = 0x00; // GP0 is active high. GP2 active low
        delay_50ms(200);
    } while (1);
```

on the processor for application use (we certainly do). Adaptor board AC162059 solves this problem – at a cost of around £20. The need of that adaptor is limited to a small range of low pin count processors, but in our view is not worth the cost. With the processor features available and the limited code space present, your applications are probably going to be simple enough that debugging with a voltmeter or oscilloscope will be sufficient. For hobbyist developments, where the speed with which you can solve problems is not critical, your £20 are certainly better spent elsewhere.

Building blocks

For a design like this, the building blocks are straight forward; for example, toggling pins, and delays of various lengths – tens of milliseconds to confirm buttons are pressed out to many tens of minutes for the power down delay. For embedded designers the creation of long delays can be problematic; the basic timing source is the processor clock oscillator, which in our case is running at 4MHz. Each instruction runs at a quarter of this rate. Creating timing delays of long lengths requires variables to count down, and spanning minutes to microseconds requires long delays and large variables. The data space for those variables is in short supply, and variables of sizes larger than eight or sixteen bits require additional software to perform these functions.

Thankfully this processor has a timer peripheral, which means we have two options: build a series of delay functions based off the time it takes individual instructions to operate, or make use of the timer peripheral that can provide slower intervals on which to build smaller functions. We will be lazy (all good software engineers are lazy!) and try the least complicated method, timing loops. If this causes issues with the amount of code required we can take the timer route.

Let's start by defining a 50ms delay function that can take a variable as an input that defines the multiple of 50ms periods for the total delay.

First we build the 'shell' of the function with some test code – toggling the LED – so we can use an oscilloscope to calibrate the delay loop. We are going to be interested in how much code space this takes up, to know if we are taking an appropriate approach to the design.

You can see the code we have added and the impact on memory utilisation in Fig.6. We have used an extra two bytes of RAM, and the program memory increased to 59 words – just 11% of available memory. That's encouraging, as most of the remaining functionality we want to add is going to be relatively simple control logic.

What you cannot see in Fig.6 are the different approaches we tested first; building a delay function that took a 32-bit integer input variable increased the program memory used to almost 100 words; here in our code we use a 16-bit integer variable, and the program takes just 59 words. The delay function has a maximum interval of just 54 minutes at a maximum input value of 65535. If you need to delay by longer, just called the function two or more times. Each subsequent call to the function adds just five words to the program, which is more efficient than using a 32-bit variable. We spent a good hour experimenting with different combinations of input variable sizes and loop delays; this seemed a reasonable comprise. 50ms delays are good for key press de-bouncing; 50 minutes seems a good time for 'lights out' or similar uses. Designing for resource constrained embedded systems can call for some compromises in how the design is achieved, and it pays to experiment with different ideas to find the best compromise for your particular needs.

Let's review Fig.6 again; looking at the 'C' program on the right, it's already beginning to get complicated, even though all this does is toggle an LED every 10s. And yet, the program takes up just 59 words of program memory. While toggling an LED is a trivial task for an NE555 timer IC, we will now move the program design into territory that the NE555 cannot go, and all within the meagre 510 words of program space available to us. Or 453 words, to be precise, after what we have created has been taken into account.

We will leave the final software development at this point until next month. Hopefully, by now you can see the potential for this device. Perhaps as a bedside light timer, or a soldering iron timer (as we use in the *PIC n' Mix* lab.)

Last month, we asked for suggestions from readers for potential projects with the PIC10F on the magazine's web forum. The feedback was great; in particular, thank you Edward Chase and Ian Stedman who came up with some excellent ideas, including safe power-down of Raspberry-Pi-based embedded systems – a major concern for industrial applications, and quite within the capabilities of this device. After we complete this month's design we will review some of these ideas and move to building real-world applications on veroboard – or possibly a professionally manufactured PCB, as this series of articles develops. And do please, keep those ideas coming!

Practical Electronics

BACK ISSUES

EVERYDAY PRACTICAL ELECTRONICS

JUNE 2018

PROJECTS • High Performance 10-Octave Stereo Graphic Equaliser – Part 1• Arduino-based Digital Inductance/ Capacitance Meter • Using Cheap Asian Electronic Modules – Part 5 & 6

FEATURES • Techno Talk • Teach-In 2018 – Get Testing! Electronic Test Equipment and Measurement Techniques – Part 9 • Net Work • PIC n' Mix • Circuit Surgery • Audio Out • Electronic Building Blocks

JULY 2018

PROJECTS • Touchscreen Appliance Energy Meter – Part 1 • Automotive Sensor Modifier • High Performance 10-Octave Stereo Graphic Equaliser – Part 2 • Using Cheap Asian Electronic Modules – Part 7

FEATURES • Techno Talk • Lascar Product Review • Net Work • PIC n' Mix • Circuit Surgery • Audio Out • Electronic Building Blocks • Lucy's Lab

AUGUST 2018

PROJECTS • Universal Temperature Alarm • Touchscreen Appliance Energy Meter – Part 2 • Banggood DSO138 LCD Scope Kit • Power Supply For Battery-Operated Valve Radios • Using Cheap Asian Electronic Modules – Part 8

FEATURES • Techno Talk • Audio Out • Lucy's Lab • Net Work • PIC n' Mix • Circuit Surgery • Electronic Building Blocks

SEPTEMBER 2018

PROJECTS • 3-Way Fully Adjustable Stereo Active Crossover For Loudspeakers – Part 1 • Dead-Easy Superhet IF Alignment Using Direct Digital Synthesis • Ultra-Low-Voltage Mini LED Flasher • Using Cheap Asian Electronic Modules – Part 9

FEATURES • Techno Talk • Net Work • PIC n' Mix • Circuit Surgery • Audio Out • Electronic Building Blocks• Lucy's Lab

OCTOBER 2018

PROJECTS • 6GHz+ Touchscreen Frequency & Period Counter – Part 1 • 3-Way Fully Adjustable Stereo Active Crossover For Loudspeakers – Part 2 • Two 230VAC Mains Timers • Using Cheap Asian Electronic Modules – Part 10

FEATURES Techno Talk • Net Work • PIC n' Mix n Circuit Surgery • Audio Out • Electronic Building Blocks • Lucy's Lab

NOVEMBER 2018

PROJECTS • Super-7 AM Radio Receiver – Part 1 • 6GHz+ Touchscreen Frequency & Period Counter – Part 2 • High-Performance Dipole Loudspeaker • Banggood's £16 100kHz-1.7GHz Build-It-Yourself SDR Kit • Dynamo Bicycle LED Lighting System

FEATURES • Techno Talk • Net Work • Lucy's Lab • Circuit Surgery • Audio Out • Electronic Building Blocks

DECEMBER 2018

PROJECTS • Touchscreen Altimeter • Super-7 AM Radio Receiver – Part 2 • 6GHz+ Touchscreen Frequency & Period Counter – Part 3 • Using Cheap Asian Electronic Modules – Part 11

FEATURES • Techno Talk • Net Work • Lucy's Lab • Teach-In 2019 – Powering Electronics – Part 1 • PIC n' Mix • Circuit Surgery • Audio Out • Electronic Building Blocks

JANUARY 2019

PROJECTS • 1.5kW Induction Motor Speed Controller – Part 1 • Altronics Mega Box • Using Cheap Asian Electronic Modules – Part 12

FEATURES • Techno Talk • Net Work • Teach-In 2019 – Powering Electronics – Part 2 • PIC n' Mix • Circuit Surgery • Max's Cool Beans • Audio Out • Electronic Building Blocks

FEBRUARY 2019

PROJECTS • 1.5kW Induction Motor Speed Controller – Part 2 • Arduino Mega Box Music Player • MIDI Ultimate Synthesiser – Part 1 • Using Cheap Asian Electronic Modules – Part 13

FEATURES • Techno Talk • Net Work • Teach-In 2019 – Powering Electronics – Part 3 • Audio Out • Circuit Surgery • PIC n' Mix • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

MARCH 2019

PROJECTS • 10-LED Bargraph • Earthquake Early Warning Alarm • Using Cheap Asian Electronic Modules – Part 14 • MIDI Ultimate Synthesiser – Part 2

FEATURES • Techno Talk • Net Work • Teach-In 2019 – Powering Electronics – Part 4 • Audio Out • Circuit Surgery • PIC n' Mix • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

APRIL 2019

PROJECTS • Internet-based 'GPS' Time Source • Infrared-sensing Heater Controller • Using Cheap Asian Electronic Modules – Part 15 • MIDI Ultimate Synthesiser – Part 3

FEATURES • Techno Talk • Teach-In 2019 – Powering Electronics – Part 5 • Net Work • PIC n' Mix • Circuit Surgery • Audio Out • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

MAY 2019

PROJECTS • 2× 12V Battery Balancer • Deluxe Frequency Switch • Using Cheap Asian Electronic Modules – Part 16 • MIDI Ultimate Synthesiser – Part 4 • USB Port Protector

FEATURES • The Fox Report • Techno Talk • Teach-In 2019 – Powering Electronics – Part 6 • Net Work • PIC n' Mix • Circuit Surgery • Audio Out • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

BACK ISSUES – ONLY £5.50

£5.50 per issue for UK incl p&p ■ £6.30 Europe Air Mail ■ £7.30 ROW Air Mail

We can supply back issues of PE/EPE by post.

We stock magazines back to 2006, except for the following:

2006 Jan, Feb, Mar, Apr, May, Jul
2007 Jun, Jul, Aug
2008 Aug, Nov, Dec
2009 Jan, Mar, Apr
2010 May, Jun, Jul, Aug, Oct, Nov
2011 Jan
2014 Jan

Issues from Jan 1999 are available on CD-ROM / DVD-ROM

If we do not have a a paper version of a particular issue, then a PDF can be supplied for the same price – your email address must be included on your order.

Please make sure all components are still available before commencing any project from a back-dated issue.

ORDER FORM – BACK ISSUES

☐ Back issues required (month / year) ....
Name ....
Address..
.... Tel: ....
Email ....
☐ I enclose cheque/PO to the value of £ ....
☐ Please charge my Visa/Mastercard £ ....
Card No ....
Valid From .... Card Expiry Date ....
Card Security Code .... (Last three digits on or under the signature strip)

SEND TO: **Practical Electronics, Wimborne Publishing Ltd**
113 Lynwood Drive, Merley, Wimborne, Dorset BH21 1UU.
Tel: 01202 880299 **Fax:** 01202 843233.
Email: stewart.kearn@wimborne.co.uk **On-line Shop**: www.epemag.com

Payments must be in £ sterling – cheque drawn on a UK bank. All items are normally posted within seven days of receipt of order. Send a copy of this form, or order by letter / online if you do not wish to cut your issue.

Max's Cool Beans

By Max the Magnificent

Options for 'micing' an Arduino

If you have read my previous columns you will know I've started to create audio-reactive projects that react to sound in general, and music in particular. If you plan on doing something like this then you need to choose a microphone. There are a number of possibilities depending on how you intend to implement your project; let's examine some options.

MAX4466-Based BOB

You can achieve some very tasty effects based only on the amplitude of the audio signal. For example, I recently supported a project called *Muzio* on Kickstarter, see: **https://bit.ly/2TQSo2H**

In this case, they are using a MAX4466-based microphone plus amplifier breakout board (BOB) with adjustable gain (via a small trimmer pot), which costs only $6.95 from Adafruit (**https://bit.ly/2svzouV**). You can supply this BOB with V_{cc} = 2.4V to 5V, so it will happily work with 3.3V and 5V MCUs. The output from the BOB is rail-to-rail, with a $V_{cc}/2$ DC bias, so you can feed it directly into one of your Arduino's analogue inputs.

ADMP401-Based BOB

Do you remember my *Awesome Audio-Reactive Artifact* project (**https://bit.ly/2I8Wktl**)? In that design I wanted to use an MSGEQ7 audio spectrum analyser chip to split the audio signal coming from the microphone into seven different frequency components,. See *Tips and Tricks for Using the MSGEQ7 Spectrum Analyzer* (**https://bit.ly/2UsaAUS**) and *Handling Noise When Working with the MSGEQ7 Audio Spectrum Analyzer* (**https://bit.ly/2FKmGza**).

The MSGEQ7 requires a $300mV_{pp}$ (peak-to-peak) audio input without any DC bias. In this case, a great choice is the ADMP401-based microphone plus amplifier BOB, which costs $10.95 from SparkFun (**https://bit.ly/1RdknR3**). This BOB puts out about $300mV_{pp}$ from a reasonably bold music source. It also has a DC bias of $V_{cc}/2$, but this is removed by the resistor-capacitor combo you add to the MSGEQ7's input discussed below.

MAX9814-Based BOB

I'm really rather pleased with the way my *Awesome Audio-Reactive Artifact* turned out, but I feel there is room for improvement. Although it works well for a reasonably bold music source, I'd like it to support a larger dynamic range and work equally well with soft and loud sound.

And so we come to the MAX9814-based microphone plus amplifier BOB with auto-gain control (AGC), which costs $7.95 from Adafruit (**https://bit.ly/2FVmMVZ**). In addition to its internal AGC capability, this little rascal also has a gain input, which can be controlled by a digital output from your Arduino. If you leave this pin unconnected (or connect it to your Arduino but set the Arduino's pin to be an INPUT), then the gain will be 60dB (ie, ×1,000). If you use the Arduino to pull this pin to 0V, the gain will be 50dB (ie, the square root of 10 × 100 ~ ×316). And if you use the Arduino to pull this pin to 5V, the gain will be 40dB (ie, ×100).

The only problem with the MAX9814-based BOB is that it puts out a $2V_{pp}$ (ie, $2,000mV_{pp}$) signal with a 1.25V DC bias. So, we somehow need to transform this $2,000mV_{pp}$ signal into the corresponding $300mV_{pp}$ signal required by the MSGEQ7.

The MSGEQ7 data sheet instructs us to use a 22kΩ resistor in series with a 100nF capacitor on the input. These values have been selected to act as a filter that passes the audio signal while removing the DC bias. Note that the reason I've named these components R2 and C2 in Fig.1(a) is to match the circuit diagrams in my aforementioned columns.

The thing with analogue is that it's a bit of a mystic art and there are typically myriad ways to do anything. One simple option is to use two resistors shown as R2a and R2b in Fig.1(b) to form a voltage divider. The trick now is to work out the values of R2a and R2b.

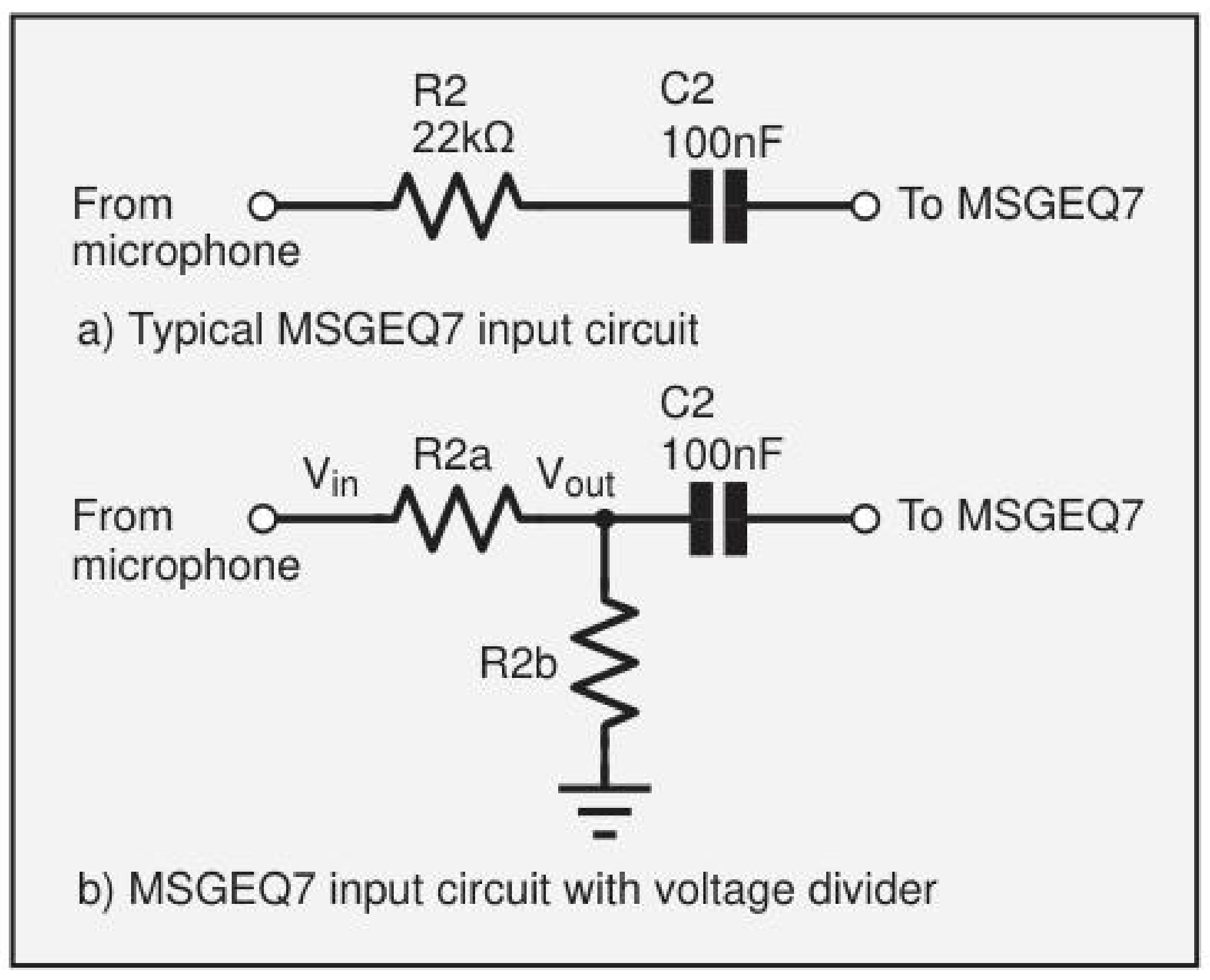


Fig.1. Modifying the input to the MSGEQ7.

Break out the maths

The formula for a voltage divider is V_{out} = (R2b / (R2a + R2b)) × V_{in}. Another way to look at this is that 2,000/300 = 6.66. So, if we think in terms of purely a unit-less ratio, then if R2b = 1, then R2a = 5.66 (if we feed these values back into our voltage divider formula, then we get $V_{out} = V_{in}/6.66$, so everything makes sense so far).

Next, we have to work out the actual values of R2a and R2b. Remember that we want the capacitor to 'see' a value of 22kΩ. The thing is that the capacitor will 'see' R2a and R2b as being in parallel, which means the resistance seen by the capacitor will be (R2a × R2b) / (R2a + R2b).

I've just been running the calculations with a variety of resistor values. Using standard 1% (E96) resistors, if R2b = 26.1kΩ, then we'd like R2a to be 5.66 × 26.1kΩ = 147.72kΩ. The nearest 1% value to this is 147kΩ. If we plug this into our 'two resistors in parallel' equation, then the capacitor will see a resistance of (26.1 × 147) / (26.1 + 147) = 22.16kΩ, which is close enough for government work.

So, what I'm thinking is that in addition to feeding the output from the microphone BOB into the voltage divider on the input of the MSGEQ7, we could also feed it directly into one of the Arduino's analogue inputs. We could then use the Arduino to periodically sample this input and keep a rolling average of its amplitude. If we see that everything has become really quiet, we could set the BOB's gain to 60dB. If the volume is increasing to the point that output from the BOB is saturating, we could reduce the gain to 50dB or 40dB. I'll let you know how things work out when I find the time to experiment with this further.

Cool bean Max Maxfield (Hawaiian shirt, on the right) is editor-in-chief at **EEWeb.com** – the go-to site for users of electronic design tools and askers of electronic questions.

Comments or questions? Email Max at: **max@CliveMaxfield.com**

Make it with Micromite

Phil Boyce – hands on with the mighty PIC-powered, BASIC microcontroller

Part 5: Traffic lights, subroutines, interrupts and IR control

This month's title may seem like a strange mix, but those topics help to demonstrate and bring together some important points regarding MMBASIC. Before I go any further, I want to reassure you that I haven't forgotten the electronic dice project. This month's article is just a little diversion that will cover concepts used in the dice mini project in the next issue.

Last month, we explained the three Micromite modes, and showed how to switch between these modes:

- Immediate (command prompt)
- Edit (program entry)
- Run (execute program)

We then examined how to connect a button (input), and an LED (output) to the *MKC* to demonstrate just how easy it is to make MMBASIC interact with these most basic of components and add some intelligence to them.

We will begin this month by quickly building a simple *Traffic Light Simulator* (*TLS*) and use several of the commands discussed last month to get it working. The *TLS* is the perfect circuit to introduce a few new MMBASIC commands, and also to learn some important (but simple) concepts. Later, we will examine how to read signals from an infrared (IR) remote transmitter; and go on to use the IR remote to control the *TLS*. Next month, we will make our first mini project – an electronic dice, and by adding the IR controller, we will create a simple prediction game.

The commands covered so far are summarised in Table 1. To recap using some of these, and also to help learn a few new commands, let's start this month by building the simple *TLS* circuit shown in Fig.1.

Traffic Light Simulator

As you can see, the *TLS* consists of three LEDs, each with a current-limiting resistor. It can easily be built on a breadboard (with jumper wires), or on a small piece of stripboard. We will not go into any construction details here as it should be very straightforward to assemble – instead, we will focus on the MMBASIC code used to add the intelligence to make the *TLS* circuit work.

The *TLS* will also be used to demonstrate a couple of important MMBASIC programming concepts: 'subroutines' and 'interrupts'; so please do follow the topics covered below. The only constructional point to note is to try and use red, yellow, and green LEDs as these will result in a more realistic simulator (rather than using three LEDs of the same colour!).

As mentioned, the *TLS* is just an LED (and resistor) connected to a digital output pin – repeated three times. Go ahead and connect the three LEDs to pins 4, 3, and 2, as shown in Fig.1.

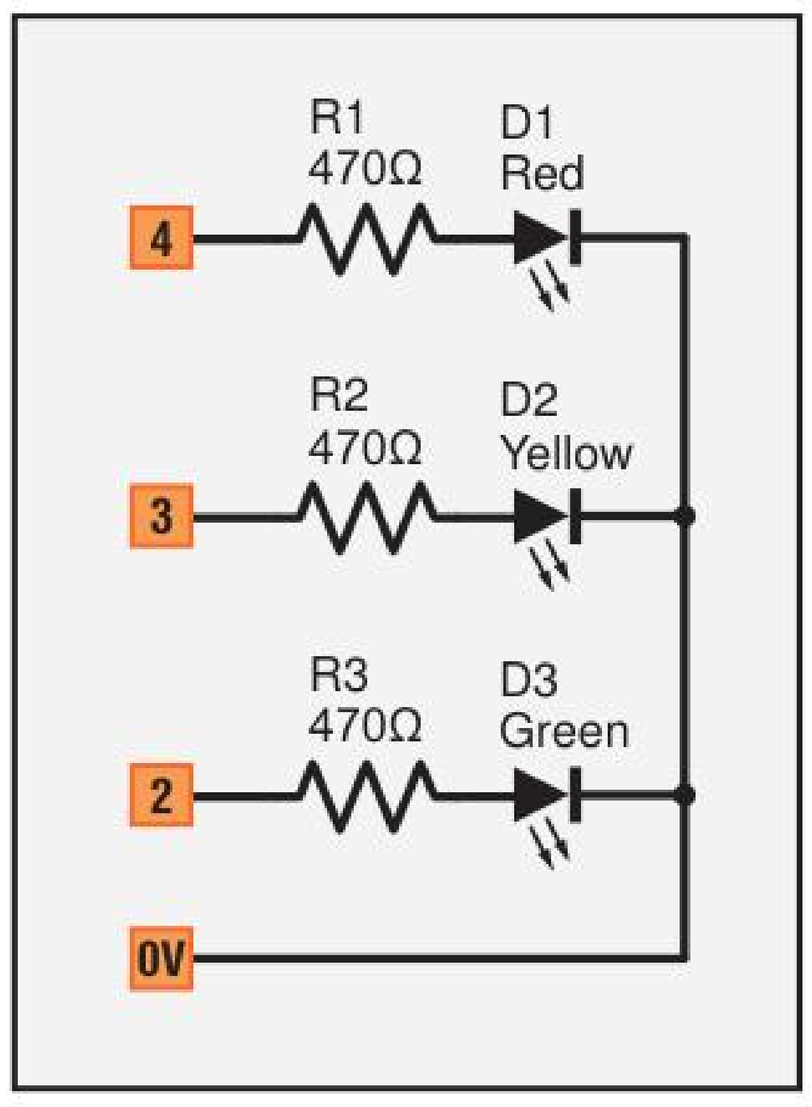


Fig.1. Circuit diagram for the *Traffic Light Simulator* (*TLS*).

Now look through each of the following lines of code. If you worked through last month's article then it should be relatively straightforward to understand what is going on (see the included comments).

Do not enter the program just yet; we will discuss next how we can 'improve it'.

```
SETPIN 4, DOUT  ' set pin 4 as a digital output
SETPIN 3, DOUT  ' pin 3 output = Amber LED
SETPIN 2, DOUT  ' pin 2 output = Green LED
D = 500         ' see text below for explanation
DO              ' defines start of loop
   PIN(4)=1     ' Red ON
   PIN(3)=0     ' amber off
   PIN(2)=0     ' green off
     PAUSE D*6  ' wait for 3000 milliseconds (3s)
```

Table 1. Summary of commands introduced in Part 4.

PRINT	Outputs to the console (TeraTerm) screen
INPUT *x*	Reads an input from the console keyboard
SETPIN(*x*), DOUT	Configures Micromite pin ***x*** to be a digital output
SETPIN(*x*), DIN	Configures Micromite pin ***x*** to be a digital input
SETPIN(*x*), DIN, PULLUP	As above, and uses an 'internal' pull-up resistor
PIN(*x*) = 0	Switches a digital output pin off (0V)
PIN(*x*) = 1	Switches a digital output pin on (+3V3)
PAUSE *x*	Waits ***x*** milliseconds (1000ms = 1s)
***labelname*:**	Defines a point (label) in a program that can be 'jumped to'
IF...THEN GOTO *labelname*	Test condition; if true then program jumps to ***labelname***
DO...LOOP	Creates a loop; code block between `DO` and `LOOP` is repeated

```
   PIN(4)=1       ' Red ON
   PIN(3)=1       ' Amber ON
   PIN(2)=0       ' green off
      PAUSE D*3   ' wait 1.5 seconds
   PIN(4)=0       ' red off
   PIN(3)=0       ' amber off
   PIN(2)=1       ' Green ON
      PAUSE D*8   ' wait 4 seconds
   PIN(4)=0       ' red off
   PIN(3)=1       ' Amber ON
   PIN(2)=0       ' green off
      PAUSE D*5   ' wait 2.5 seconds
LOOP              ' loop back to DO
```

Something new introduced in the code above is the fourth line: `D = 500`

In a BASIC program you will usually use one or more 'variables'. A variable contains a numeric value and it can be referenced anywhere in a program to 'recall' the value stored in it. In the example above, the variable has been given the reference (ie, name) `D`. A variable's name needs to be unique, and it cannot be the same as an existing MMBASIC command (ie, we could not use `PRINT = 500`). Note that, as the name implies, a variable can have its value changed (by the program) and hence it does not need to remain constant (hence 'variable'!). However, in the *TLS* program, the value is not altered anywhere by the code.

In our program, the `PAUSE` lines use the value stored in the variable `D` (ie, `500`) to work out how many milliseconds (ms) the program should wait. So `PAUSE D` is equivalent to `PAUSE 500`, which causes the program to wait for 500ms or half a second; and `PAUSE D*6` for example, will pause for (500 × 6) ms = 3000ms or three seconds. An alternative way to look at this is to think of the 'multiplier' of `D` (×6 in this last example) as the number of 'half seconds' (if `D = 500`) that the program will pause for.

The reason we are using a variable here in our program is that if we want to alter the speed of the *TLS* sequencing, then we only need to change a single line of code. So if we were to change the fourth line of code (above) to `D = 1000` then the *TLS* will run slower (at half the speed); and changing it to `D = 250` will make it run twice as fast.

With MMBASIC, multiple commands can be put onto a single line of code. To do this we simply use a ':' (colon) between the commands. Doing this not only shortens the overall length of the program (by reducing the total number of lines), but it can also make the code easier to follow. The shortened (12-line) version of the 22-line *TLS* program above is as follows (most comments have also been removed to make program entry quicker):

```
SETPIN 4, DOUT : SETPIN 3, DOUT : SETPIN 2, DOUT
D = 500
DO
   PIN(4)=1 : PIN(3)=0 : PIN(2)=0    ' red LED on
      PAUSE D*6
   PIN(4)=1 : PIN(3)=1 : PIN(2)=0    ' red + amber
      PAUSE D*3
   PIN(4)=0 : PIN(3)=0 : PIN(2)=1    ' green
      PAUSE D*8
   PIN(4)=0 : PIN(3)=1 : PIN(2)=0    ' amber
      PAUSE D*5
LOOP
```

Now enter this shortened version of the code. Connect your *MKC* and *DM* to your computer, and then start TeraTerm. Remember to select the relevant COM port that Windows assigns to the *DM*. Next, type `NEW` (at the command prompt) to remove any existing code, and then press F4 to enter EDIT mode. You can then type in the program – the comments are not essential.

When you have finished entering all 12 lines, press F2 to Save and `RUN` your program. If all is correct, then you should see the LEDs simulate a *TLS* sequence, pausing longer on red, and on green (as in real life). If anything doesn't perform as expected, then its time to troubleshoot. There is not much to go wrong, but if something isn't working then check the following hardware: resistor values (between 330Ω and 1kΩ), LED orientation (remember – LEDs are polarised), 0V line connected, and correct points used to connect to pins 4, 3, and 2. With regards to software, check: correct pin numbers entered, no error messages displayed when the program is `RUN` (if so – then a clue will be provided in the error message), and check that all 12 code lines are present and correctly entered.

The `PORT` command

We will now introduce a new command, `PORT`. This command is used to set *multiple* digital outputs in a single command. We will also show how to use a binary value to help switch the multiple digital outputs on or off. In our program above, consider the line of code (immediately below the `DO` command):

```
PIN(4)=1 : PIN(3)=0 : PIN(2)=0 ' red LED on
```

The result is that it sets multiple digital outputs either on or off. This line of code actually contains three separate commands (each separated by a '`:`'). We are setting multiple (ie, three) digital outputs that are connected to pins 2, 3, and 4; switching each output either on (with `1`) or off (with `0`).

These three commands can be replaced by a single `PORT` command, which has the syntax (structure): `PORT(x,y)=z`

Here, `x` represents the 'starting' pin, `y` equals the total number of sequential pins, and `z` represents the data (value) to output. Now this may seem confusing at first, so let's expand on this with the actual single `PORT` command that can replace the three separate `PIN()` commands above:

```
PORT(2,3)=&b100 ' red
```

Here, the instruction is referencing the three (`y` = 3) numerically sequential pins starting at pin 2 (`x` = 2), in other words, pins 2, 3 and 4. The `&b100` value that we are using is in binary notation (indicated by the prefix `&b`). Don't worry if you know nothing about binary – for now, just accept that binary numbers comprise only of digits '0' and '1'; and for our purpose here, the binary digits usefully represent 'off' (0) or 'on' (1). So the three-digit binary value: `&b100` represents: 'on', 'off', 'off'.

An important point to note here is that the value's right-hand binary digit (ie, `0`) is applied to the 'starting pin' defined in the `PORT` command (ie, pin 2), and then each subsequent binary digit in the value (working over to the left-most digit) is then applied to subsequent pins following the starting pin. Thus, pin 2 = off, pin 3 = off, and pin 4 = on.

If you are still confused, then just consider the three-digit binary value to indicate the on (1) or off (0) states applied sequentially to the red, amber, and green LEDs. So `100` will turn the red LED (pin 4) On (`1`), the amber LED (pin 3) off (`0`), and the green LED (pin 2) off (`0`).

A more detailed explanation of using the `PORT` command can be found in the *User Manual* if you want to explore things further.

Now update the *TLS* program by replacing the four relevant lines of code (containing the multiple `PIN()` commands) with the following four `PORT` commands:

```
PORT(2,3)=&b100 ' red
PORT(2,3)=&b110 ' red + amber
PORT(2,3)=&b001 ' green
PORT(2,3)=&b010 ' amber
```

When you `RUN` your program, nothing will appear to have changed – this is good since all we did here was to make the code more compact and a little easier to read!

These last few examples of the *TLS* program demonstrate an important point made in *Part 4* – '… there are many ways to achieve the same end result. No one way is right or wrong; but some ways may be better than others'.

Subroutines

A 'subroutine' is a very useful concept in coding, and will be explained here by applying it to our *TLS* program.

Most programs can be split into three sections:

- Startup (to configure various things)
- Main program (easy-to-read program flow)
- Subroutines (one or more detailed chunks of code that perform specific tasks)

With our *TLS* program, everything is currently grouped together as one main program flow. Now, consider the following code (which produces the same end result):

```
' startup
SETPIN 4, DOUT : SETPIN 3, DOUT
SETPIN 2, DOUT
D = 500

' main program
DO
   TL_Red
   TL_RedAmber
   TL_Green
   TL_Amber
LOOP

' subroutines
SUB TL_Red
   PORT(2,3)=&b100
   PAUSE D*6
END SUB

SUB TL_RedAmber
   PORT(2,3)=&b110
   PAUSE D*3
END SUB

SUB TL_Green
   PORT(2,3)=&b001
   PAUSE D*8
END SUB

SUB TL_Amber
   PORT(2,3)=&b010
   PAUSE D*5
END SUB
```

Here, we have the same startup section (the `SETPIN`s and the D=500) followed by the main program highlighted in bold. The main program has now been made much easier to read (bear with me!) by implementing four subroutines (SUBs).

`TL_Red` is the name given here to a detailed chunk of code (ie, a subroutine) that performs the specific task of switching the red LED on, and the amber and green LEDs off. The three other `SUB`s have also been given unique and meaningful names that provide a good clue as to what the subroutine code does.

There are two commands associated with creating a subroutine as can be seen in the above code listing:

`SUB` ***`subname`*** – goes at the start
   `(subroutine code here)`
`END SUB` – placed at the end

Even though in this particular example the overall length of the program has increased, it is now much easier to follow. Not only that, if we are repeating a task (with the exact same code) in our main program, then implementing the task code using a `SUB` means that any changes to that task requires changing code in one location only; ie, changing the `SUB` code rather than changing multiple lines of code in the main program.

So in a program containing `SUB`s, the program flow will work down from the beginning, but will divert off to the specific subroutine code whenever a *`subname`* is encountered. Once the end of the subroutine code has been reached, `END SUB` will return the program flow back to the line of code immediately after the *`subname`*.

For more details regarding subroutines, refer to the *User Manual* (*Defined Subroutines and Functions* section on page 43 in the current version).

Interrupts

When a program is `RUN`, the program flow will always start at the first line of code, and will then methodically work down through the subsequent lines of code. As we have just seen, the program flow may divert off to process a subroutine; or if a certain condition is tested (with `IF... THEN GOTO` *`labelname`*) then program flow may jump to a different part of the program (ie, to *`labelname`*`:`). However, what if we immediately wanted our code to do something when a particular physical event occurs – such as sound an alarm when a panic button is pushed. This is where interrupts come in because they are given priority in controlling program flow.

The source of an interrupt (ie, the trigger) can be from various things. One example is a digital input pin sensing a change of voltage (ie, going from +V to 0V) because a button is pressed. The interrupt has to be set up before it will respond to the trigger; and once set up (and triggered), then the end result is that a subroutine (*`subname`* defined in the interrupt set up) is immediately called. In this case the subroutine is referred to as an 'interrupt `SUB`', but in terms of coding, it looks like any other type of `SUB`.

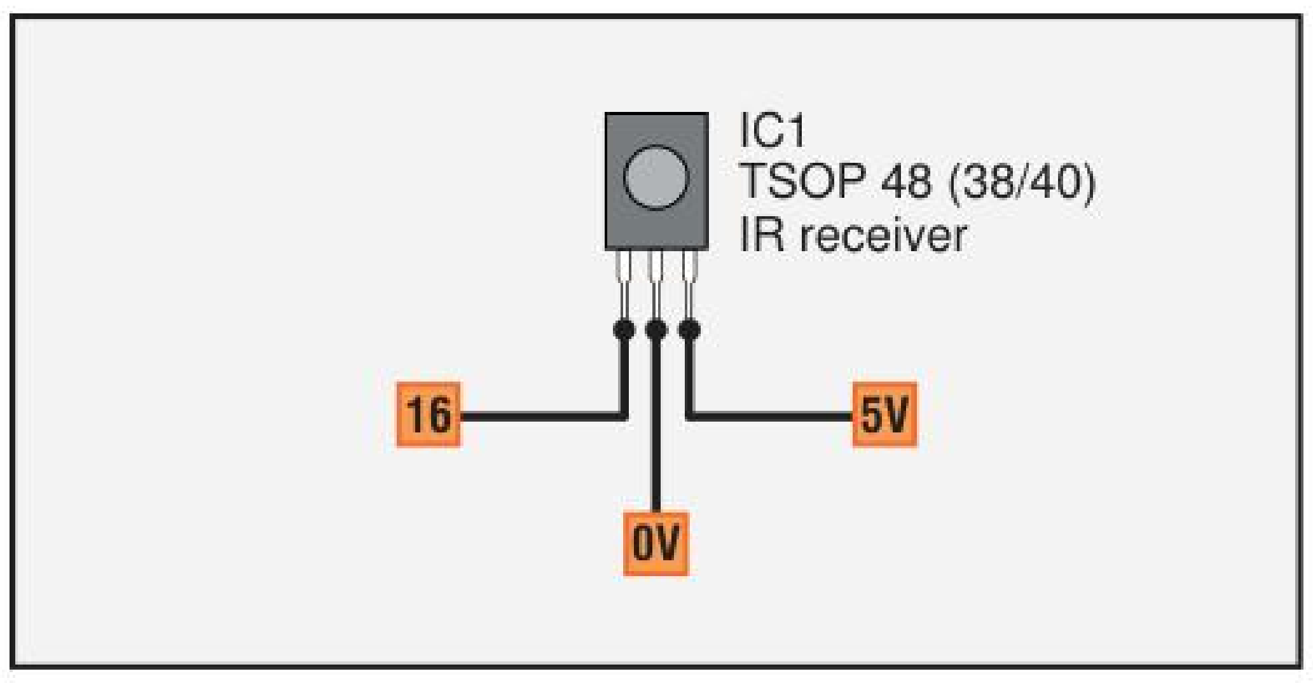


Fig.2. How to connect the IR-receiver. It will only work on Pin 16 of the *MKC*.

Another source of interrupt is an IR signal received on Pin 16. We will now explore this further as it helps to nicely demonstrate the use of `SUB`s and interrupts. It will also allow us to easily add IR control to the *TLS* program.

Adding an IR Remote control

First, we need to highlight some IR basics. Most IR remotes transmit two pieces of information (ie, codes) when a button is pressed. These two codes are the:

- Device code – a number that identifies the remote controller
- Key Code – a number that identifies which button (key) is being pressed.

MMBASIC makes it extremely easy to extract these two numeric codes from a remote control. All you need is a single component (an IR receiver) and just a few lines of code.

IR transmitters transmit infrared (invisible light) at either 38kHz or 40kHz (but variations to these do exist). A suitable IR receiver is one from the commonly available TSOP4xxx series. So a TSOP4x38 (38kHz) or a TSOP4x40 (40kHz) is ideal to start exploring IR codes.

Let's now add IR capability to the *TLS*. To connect an IR receiver to your *MKC*, refer to Fig.2. Pin 16 on the Micromite is a special pin which can perform various tasks, including reading the data from an IR receiver. Now make the three connections as shown. (Note that the IR receiver requires a 5V supply. It can connect to the 4-way headers J7 (0v) and J9 (+5V) on the *DM*.)

To enable pin 16 to respond to IR signals, we need to set up an interrupt. This is very easily achieved with the following single line of code (placed ideally anywhere in the start-up section of a program):

```
IR DevCode, KeyCode, IR_Int
```

What this effectively does is switch on IR decoding within the Micromite (when the program is run). Then, whenever a valid IR signal is detected, MMBASIC will automatically load two variables (that we have named `DevCode` and `KeyCode`) with the Device Code and the Key Code numbers that are contained within the IR signal.

The `IR_Int` part of the command is simply defining the name of the interrupt `SUB` that is immediately called whenever a valid IR signal has been detected. In this interrupt subroutine we can use the values received into `DevCode` and `KeyCode` to make our program do whatever is required.

Let's now add the following interrupt `SUB` to the very end of our *TLS* program.

```
SUB IR_Int
   PRINT "Remote ID = " DevCode
   PRINT "Key Code = " KeyCode
END SUB
```

All this does (for now) is display the `DevCode` and `KeyCode` values returned whenever a button is pressed on the IR remote. The information is displayed on the console screen (ie, in TeraTerm) because we are using the `PRINT` command. Don't worry if the `PRINT` command seems confusing – for now, just copy it exactly as shown.

If you have not purchased the *IR Remote Kit* from **micromite.org** then simply use any remote that you likely have at home (TV, Hi-Fi, home cinema...) – you will off course still need to obtain the IR receiver. Note that some manufactures use their own custom IR signals – so it is handy if you have several remotes available to try.

When you run the program, the program flow will still be the same and hence the *TLS* will work as before. But now, when you press a button on an IR remote you should see some numbers appear in TeraTerm (if not – first try different remotes, then check connections are as shown in Fig.2).

Press different buttons on the remote – you should see that the `DevCode` (ie, the Remote ID) value remains the same, however, the `KeyCode` values will vary.

Now, we will use the IR remote to control the *TLS*. We will modify the program to make the *TLS* wait on the red signal, and it will only start sequencing once a particular button is pressed. The sequence will continue until any other button is pressed, at which point it will wait on red again (after first completing the current sequence back to red).

With the code running, chose any button on the remote, and write down the `DevCode` and `KeyCode` values returned when your chosen button is pressed (on my remote the values are 255 and 56 respectively).

Now modify the interrupt `SUB` to become the following:

```
SUB IR_Int
   BP=0
   ' substitute with your two values
   If DevCode=255 And KeyCode=56 Then BP=1
   PRINT "Remote ID = " DevCode
   PRINT "Key Code = " KeyCode " (BP=" BP ")"
End SUB
```

What we are doing here is setting the value of a variable (we've named `BP`) to either `0` or `1`. It will be set to `1` whenever our chosen button is pressed on the IR remote. Any other button will leave `BP` set to a value of `0`. We refer to `BP` as a 'flag' to indicate if our chosen button has been pressed.

In the above modified interrupt `SUB`, we are also showing the value of `BP` on the console. `RUN` the program again. You should see that `BP=0` for every button apart from your chosen button (when you'll see that `BP=1`). Ensure that this works as expected (making any changes as needed).

Now that we have the `BP` flag set to a value of `0` or `1`, we can check its value in our main program to alter what happens.

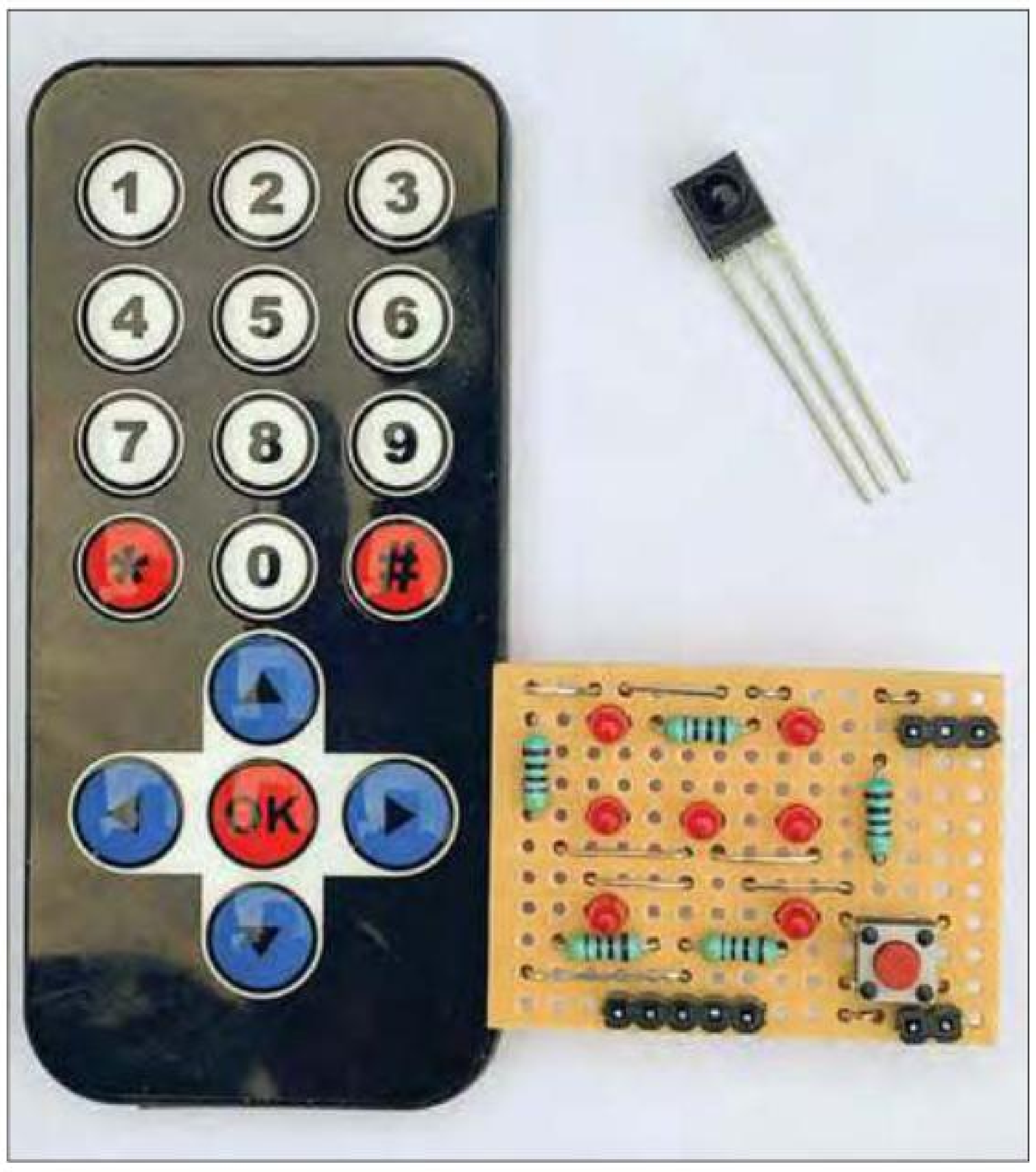


Fig.3. We can use a simple remote control (IR transmitter) and an IR receiver (top right) to trigger a Micromite interrupt.

We will use this capability in the electronic dice project next month.

Remember, we want to have the *TLS* wait on red, and only start the sequence if the chosen button is pressed. So we just need to add a single line of code (highlighted in bold) to the main program, as shown below:

```
' main program
DO
   TL_Red
   DO : LOOP UNTIL BP = 1
   TL_RedAmber
   TL_Green
   TL_Amber
LOOP
```

It should be self-explanatory what is happening here – note how easy it is to read the program flow because everything is structured into sections.

Now `RUN` your program and check it works as expected. Wait a few seconds before pressing your chosen button – you should then see it start sequencing as soon as you press it. The code will continue to sequence the *TLS* until you press another button. When you do press any other button, note that the current sequence will complete first (back to red) – then it will wait once again for the chosen button to be pressed.

One last thing to see; reduce the value of `D` (by changing the fourth line to `D = 20`) and then `RUN` your program again. This time, IR button presses will appear much more 'responsive', as there is no long wait for the sequence to 'finish'.

Hopefully, you can now see that it is possible to add all kinds of logic and intelligence to the *TLS* program.

For example, have a go at using one button to sequence from red to green; and then a second button to sequence back to red. Hint: you will need to add just two lines of code to the program:

- One line into the interrupt `SUB` – (another `IF...` setting `BP` to a value of `2`)
- One line (`DO: LOOP UNTIL...`) into the main program (after `TL_Green`) checking `BP` for the value of `2`

This has nicely demonstrated how easy it is to use `SUB`s and interrupts; and along the way we have learnt how to hook up an IR remote to a project. With the MMBASIC building blocks you have learnt so far you can hopefully begin to realise that lots of possibilities are opening up when using a Micromite as the 'brains' inside your projects.

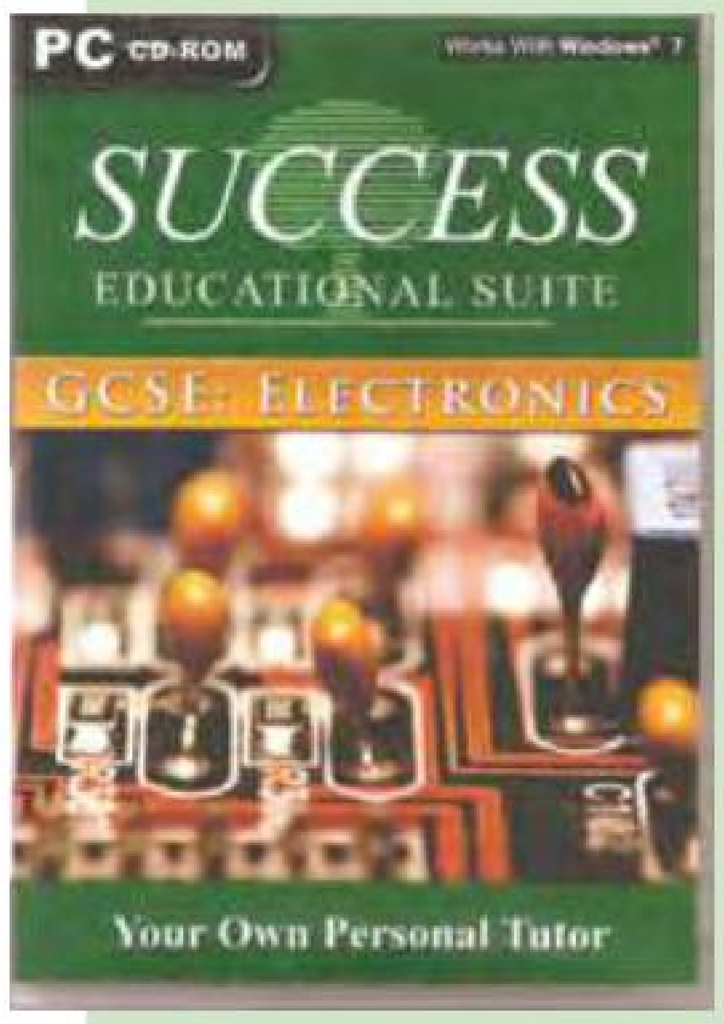
Next month, we will learn more MMBASIC commands, and build the electronic dice.

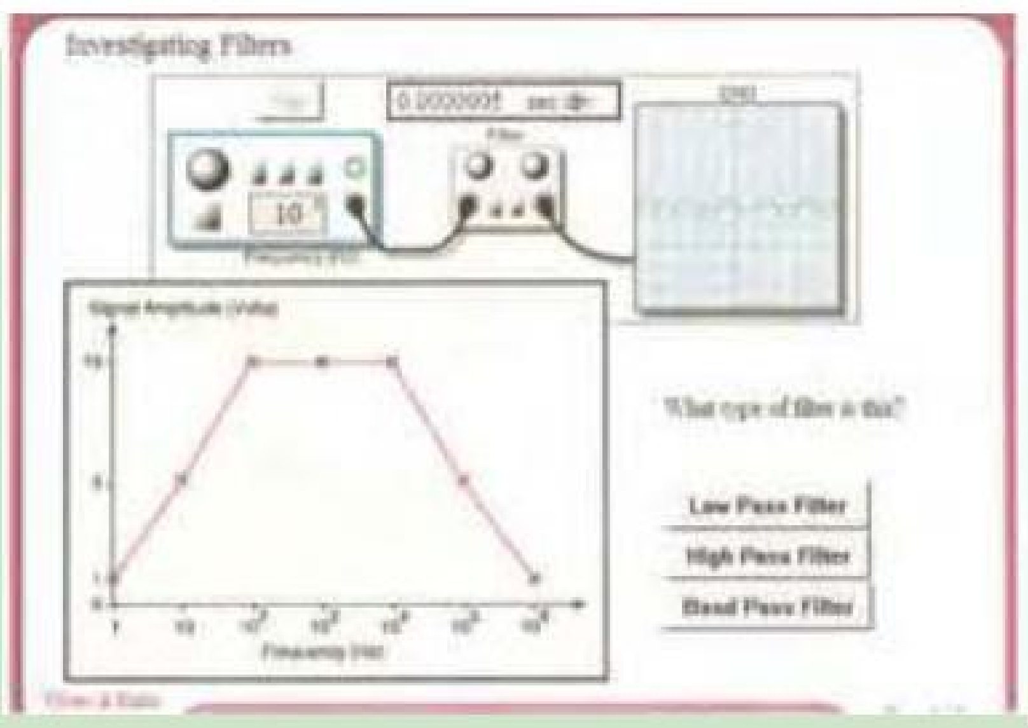
Questions? Please email Phil at:
contactus@micromite.org

PRACTICAL ELECTRONICS IS PLEASED TO OFFER YOU THESE

ELECTRONICS CD-ROMS

GCSE ELECTRONICS





Suitable for any student who is serious about studying and who wants to achieve the best grade possible. Each program's clear, patient and structured delivery will aid understanding of electronics and assist in developing a confident approach to answering GCSE questions. The CD-ROM will be invaluable to anyone studying electronics, not just GCSE students.

*** Contains comprehensive teaching material to cover the National Curriculum syllabus * Regular exercises reinforce the teaching points * Retains student interest with high quality animation and graphics * Stimulates learning through interactive exercises * Provides sample examination ques-tions with model solutions * Authored by practising teachers * Covers all UK examination board syllabuses * Caters for all levels of ability * Useful for self-tuition and revision**

£12.50
inc. VAT and P&P

SUBJECTS COVERED

Electric Circuits – Logic Gates – Capacitors & Inductors – Relays – Transistors – Electric Transducers – Operational Amplifiers – Radio Circuits – Test Instruments

Over 100 different sections under the above headings

CIRCUIT WIZARD

Circuit Wizard is a revolutionary software system that combines circuit design, PCB design, simulation and CAD/CAM manufacture in one complete package. Two versions are available – Standard or Professional.

By integrating the entire design process, Circuit Wizard provides you with all the tools necessary to produce an electronics project from start to finish – even including on-screen testing of the PCB prior to construction!

* Circuit diagram design with component library (500 components Standard,1500 components Professional) * Virtual instruments (4 Standard, 7 professional)* On-screen animation * Interactive circuit diagram simulation * True analogue/digital simulation * Simulation of component destruction * PCB Layout * Interactive PCB layout simulation * Automatic PCB routing * Gerber export * Multi-level zoom (25% to 1000%) * Multiple undo and redo * Copy and paste to other software * Multiple document support

This software can be used with the *Jump Start* and *Teach-In 2011* series (and the *Teach-In 4* book).

Standard **£61.25** inc. VAT. Professional **£75** plus VAT.

From £49.00

TINA Design Suite V11

FEATURED IN OUR TEACH-IN 2015 SERIES

Analogue, Digital, Symbolic, RF, MCU and Mixed-Mode Circuit Simulation and PCB Design with TINA

TINA Design Suite V11 is a powerful yet affordable software package for analysing, designing and real time testing analogue, digital, MCU, and mixed electronic circuits and their PCB layouts. You can also analyse RF, communication, optoelectronic circuits, test and debug microcontroller applications.

Enter and analyse any circuit up to 100 nodes (student), or 200 with the Basic (Hobbyist) version within minutes with TINA's easy-to-use schematic editor. Enhance your schematics by adding text and graphics. Choose components from the large library containing more than 10,000 manufacturer models. Analyse your circuit through more than 20 different analysis modes or with 10 high tech virtual instruments.

Present your results in TINA's sophisticated diagram windows, on virtual instruments, or in the live interactive mode where you can even edit your circuit during operation.

Customise presentations using TINA's advanced drawing tools to control text, fonts, axes, line width, colour and layout. You can create and print documents directly inside TINA or cut and paste your results into your favourite word procesing or DTP package.

TINA includes the following Virtual Instruments: Oscilloscope, Function Generator, Multimeter, Signal Analyser/Bode Plotter, Network Analyser, Spectrum Analyser, Logic Analyser, Digital Signal Generator, XY Recorder.

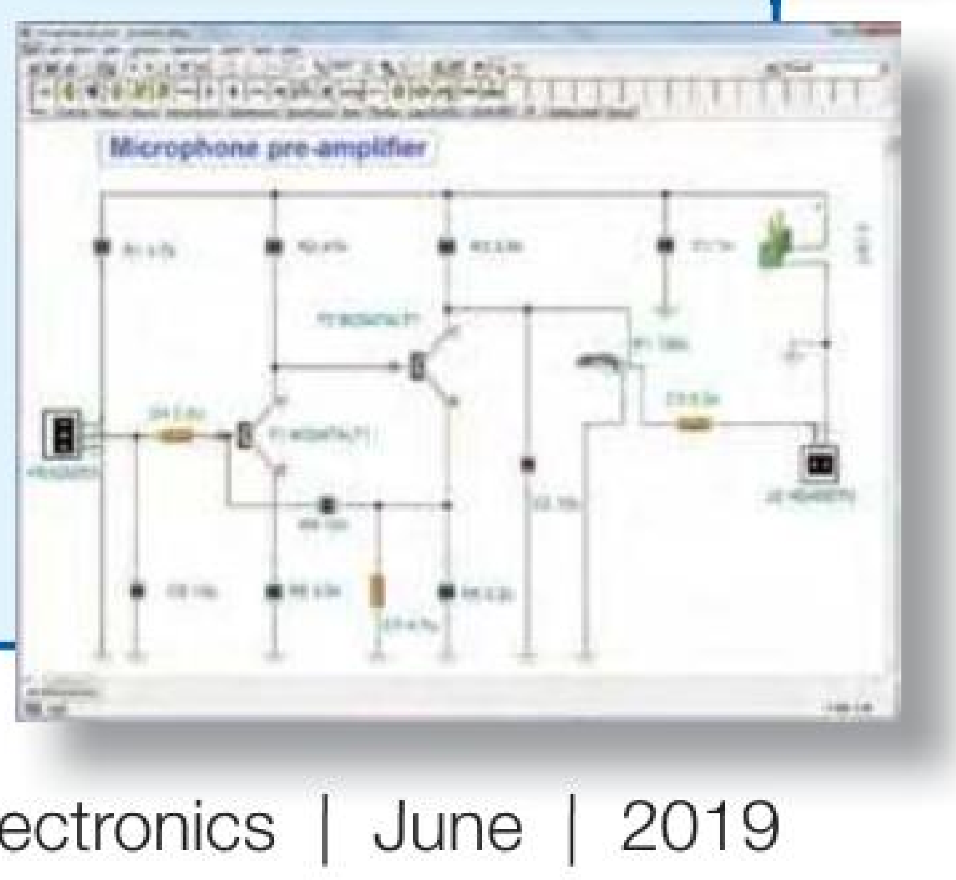
3D Viewer
File View Options

This offer gives you a CD-ROM – the software will need registering (FREE) with Designsoft (TINA), details are given within the package.

Get TINA Basic V11 (Hobbyist) for £129 or Student V11 version for £49

Prices include VAT and UK postage

+ get a 1 year free subscription for TINACloud the breakthrough cloud version of TINA which you can run on most operating systems and computers, including PCs, Macs, thin clients iPads and other tablets – even on many smart phones, smart TVs and e-book readers.



To order please either fill out and return the order form, or call us on 01202 880299

Alternatively you can order via our secure online shop at: www.epemag.com

PICmicro TUTORIALS AND PROGRAMMING

HARDWARE

PICmicro Multiprogrammer Board and Development Board

Suitable for use with the three software packages listed below

This flexible PICmicro microcontroller programmer board and combination board allows students and professional engineers to learn how to program PICmicro microcontrollers as well as program a range of 8, 18, 28 and 40 pin devices from the 12, 16 and 18 series PICmicro ranges. For those who want to learn, choose one or all of the packages below to use with the hardware.

- Makes it easier to develop PICmicro projects
- Supports low cost Flash-programmable PICmicro devices
- Fully featured integrated displays – 16 individual LEDs, quad 7-segment display and alphanumeric LCD display
- Supports PICmicro microcontrollers with A/D converters
- Fully protected expansion bus for project work
- USB programmable
- Compatible with the E-blocks range of accessories

£118 including VAT and postage

SOFTWARE

ASSEMBLY FOR PICmicro V6

(Formerly PICtutor)

Assembly for PICmicro microcontrollers V3.0 (previously known as PICtutor) by John Becker contains a complete course in programming the PIC16F84, 16F88 and 16F877a PICmicro microcontroller from Arizona Microchip. It starts with fundamental concepts and extends up to complex programs including watchdog timers, interrupts and sleep modes.

The CD makes use of the latest simulation techniques which provide a superb tool for learning: the Virtual PICmicro microcontroller, this is a simulation tool that allows users to write and execute MPASM assembler code for the PIC16F84 microcontroller on-screen. Using this you can actually see what happens inside the PICmicro MCU as each instruction is executed, which enhances understanding.

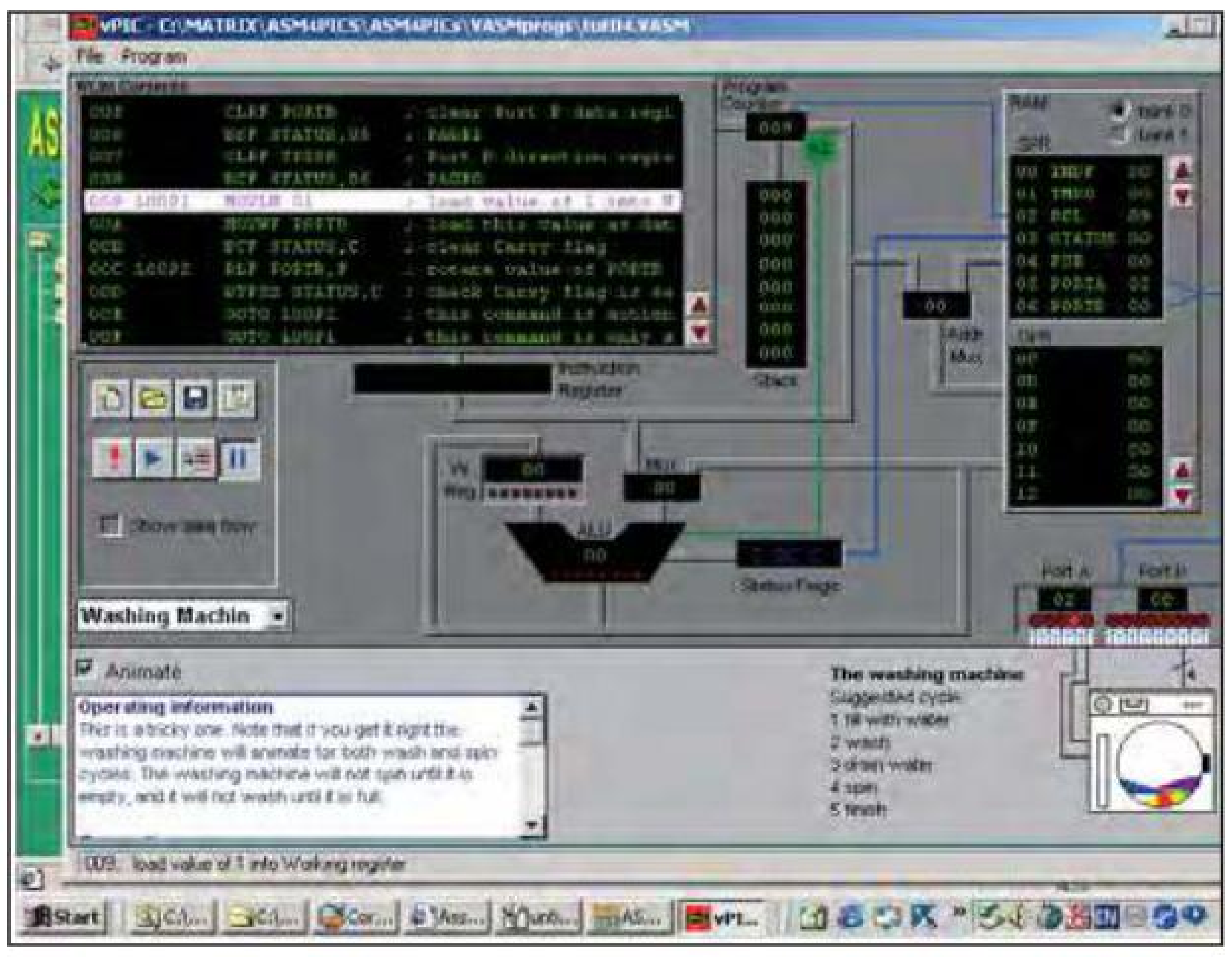
- Comprehensive instruction through 45 tutorial sections
- Includes Vlab, a Virtual PICmicro microcontroller: a fully functioning simulator
- Tests, exercises and projects covering a wide range of PICmicro MCU applications
- Includes MPLAB assembler
- Visual representation of a PICmicro showing architecture and functions
- Expert system for code entry helps first time users
- Shows data flow and fetch execute cycle and has challenges (washing machine, lift, crossroads etc.)
- Imports MPASM files.

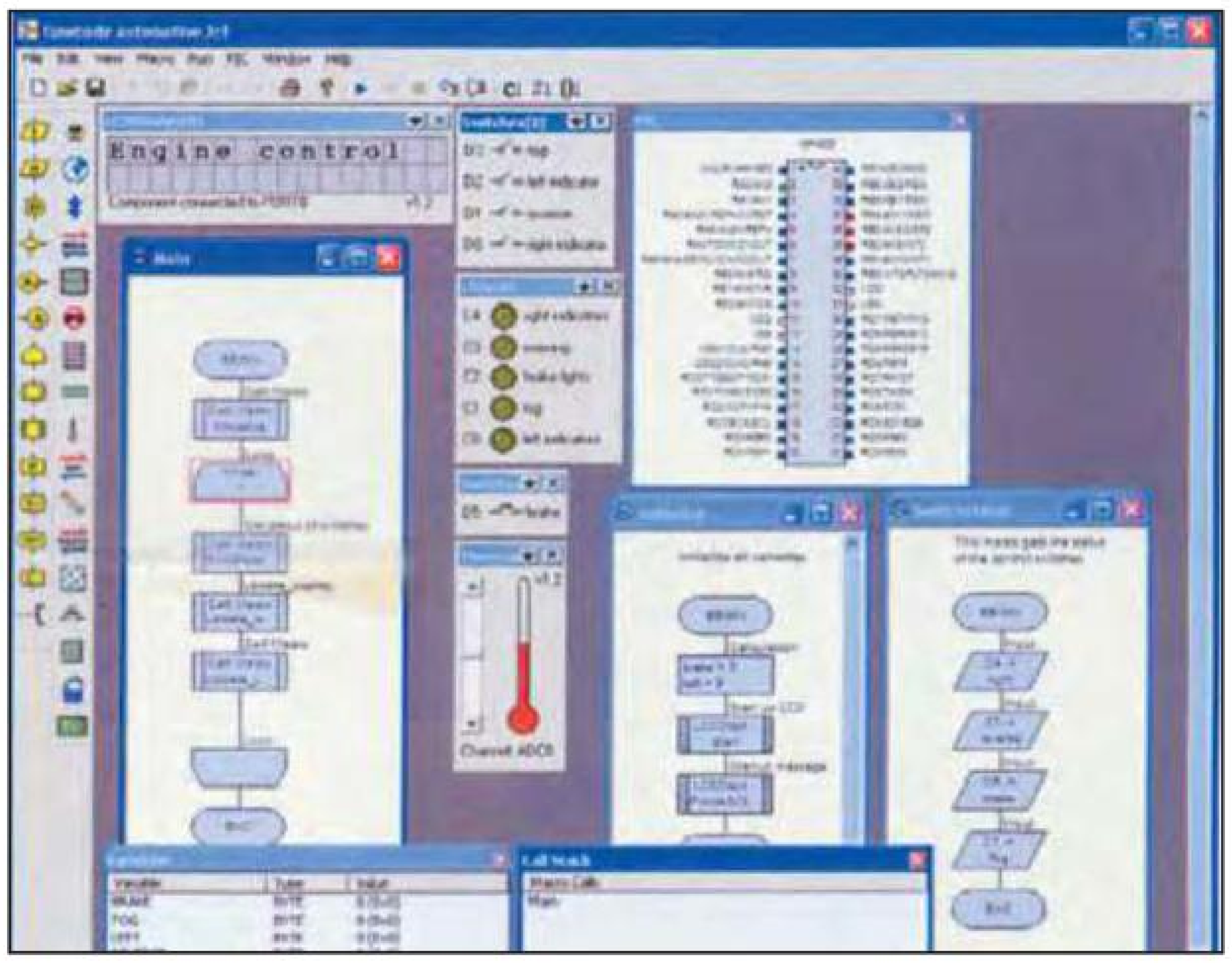
FLOWCODE FOR PICmicro V8

Flowcode is a very high level language programming system based on flowcharts. Flowcode allows you to design and simulate complex systems in a matter of minutes. A powerful language that uses macros to facilitate the control of devices like 7-segment displays, motor controllers and LCDs. The use of macros allows you to control these devices without getting bogged down in understanding the programming. When used in conjunction with the development board this provides a seamless solution that allows you to program chips in minutes.

- Requires no programming experience
- Allows complex PICmicro applications to be designed quickly
- Uses international standard flow chart symbols
- Full on-screen simulation allows debugging and speeds up the development process.
- Facilitates learning via a full suite of demonstration tutorials
- Produces code for a wide range of devices
- 16-bit arithmetic strings and string manipulation
- Pulse width modulation
- I2C.

Please note: Due to popular demand, Flowcode is now available as a download. Please include your email address and a username (of your choice) on your order. A unique download code will then be emailed to you.





PRICES

Prices for each of the CD-ROMs above are:

(Order form on next page)

(UK and EU customers add VAT to ‘plus VAT’ prices)

Single License . . . . £99 plus VAT

Site Licence . . . . £499 plus VAT

Flowcode . . . . Contact us for pricing
(choose PIC-8b, PIC-16b, PIC-32b, AVR/Arduino,ARM)

ELECTRONICS TEACH-IN 2

ELECTRONICS TEACH-IN 2 CD-ROM
USING PIC MICROCONTROLLERS A PRACTICAL INTRODUCTION

This *Teach-In* series of articles was originally published in *EPE* in 2008 and, following demand from readers, has been collected together in the *Electronics Teach-In 2* CD-ROM.

The series is aimed at those using PIC microcontrollers for the first time. Each part of the series includes breadboard layouts to aid understanding and a simple programmer project is provided.

Also included are 29 *PIC N' Mix* articles, also republished from *EPE*. These provide a host of practical programming and interfacing information, mainly for those that have already got to grips with using PIC microcontrollers. An extra four part beginners guide to using the C programing language for PIC microcontrollers is also included.

The CD-ROM also contains all of the software for the *Teach-In 2* series and *PIC N' Mix* articles, plus a range of items from Microchip – the manufacturers of the PIC microcontrollers. The material has been compiled by Wimborne Publishing Ltd. with the assistance of Microchip Technology Inc.

CD-ROM **Order code ETI2 CD-ROM** **£9.50**

ELECTRONICS TEACH-IN 3

ELECTRONICS TEACH-IN 3 CD-ROM

The three sections of this CD-ROM cover a very wide range of subjects that will interest everyone involved in electronics, from hobbyists and students to professionals. The first 80-odd pages of Teach-In 3 are dedicated to *Circuit Surgery*, the regular *EPE* clinic dealing with readers' queries on circuit design problems – from voltage regulation to using SPICE circuit simulation software.

The second section – *Practically Speaking* – covers the practical aspects of electronics construction. Again, a whole range of subjects, from soldering to avoiding problems with static electricity and indentifying components, are covered. Finally, our collection of *Ingenuity Unlimited* circuits provides over 40 circuit designs submitted by the readers of *EPE*.

The CD-ROM also contains the complete *Electronics Teach-In 1* book, which provides a broad-based introduction to electronics in PDF form, plus interactive quizzes to test your knowledge, TINA circuit simulation software (a limited version – plus a specially written TINA Tutorial).

The Teach-In 1 series covers everything from Electric Current through to Microprocessors and Microcontrollers and each part includes demonstration circuits to build on breadboards or to simulate on your PC.

CD-ROM **Order code ETI3 CD-ROM** **£8.50**

ELECTRONICS TEACH-IN 4

ELECTRONICS TEACH-IN 4 CD-ROM
A BROAD-BASED INTRODUCTION TO ELECTRONICS.

The Teach-In 4 CD-ROM covers three of the most important electronics units that are currently studied in many schools and colleges. These include, Edexcel BTEC level 2 awards and the electronics units of the Diploma in Engineering, Level 2.

The CD-ROM also contains the full Modern Electronics Manual, worth £29.95. The Manual contains over 800 pages of electronics theory, projects, data, assembly instructions and web links.

A package of exceptional value that will appeal to all those interested in learning about electronics or brushing up on their theory, be they hobbyists, students or professionals.

CD-ROM **Order code ETI4 CD-ROM** **£8.99**

ELECTRONICS TEACH-IN 5

FREE CD-ROM
£8.99
ELECTRONICS TEACH-IN 5
JUMP START
PRACTICALLY SPEAKING
PIC N MIX
TEACH-IN 2
TWO TEACH-INs FOR THE PRICE OF ONE
Electronics Teach-In 2

ELECTRONICS TEACH-IN 5
JUMP START

FREE CD-ROM

15 design and build circuit projects dedicated to newcomers or those following courses in school and colleges. The projects are: Moisture Detector, Quiz Machine, Battery Voltage Checker, Solar-Powered Charger, Versatile Theft Alarm, Spooky Circuits, Frost Alarm, Mini Christmas Lights, iPod Speaker, Logic Probe, DC Motor Controller, Egg Timer, Signal Injector Probe, Simple Radio Receiver, Temperature Alarm.

PLUS:

PIC' N MIX – starting out with PIC Microcontrollers and PRACTICALLY SPEAKING – the techniques of project construction.

FREE CD-ROM – The free CD-ROM is the complete Teach-In 2 book providing a practical introduction to PIC Microprocessors plus MikroElektronika, Microchip and L-Tek PoScope software.

160 Pages **Order code ETI5** **£8.99**

ELECTRONICS TEACH-IN BUNDLE – FOR PARTS 3, 4 & 5

CD-ROM **Order code ETIB3 CD-ROM** **£18.95**

ORDERING
ALL PRICES INCLUDE UK POSTAGE

Standard/Student/Basic (Hobbyist) Version price includes postage to most countries in the world EU residents outside the UK add £5 for airmail postage per order

Please send me: CD-ROM ORDER FORM

☐ **Assembly for PICmicro V6**
☐ **'C' for 16 Series PICmicro V6**

Version required:
☐ **Single Licence**
☐ **Site licence**

Note: The software on each version is the same, only the licence for use varies.

☐ **PICmicro Multiprogrammer Board and Development Board (hardware)**

☐ **Circuit Wizard – Standard**
☐ **Circuit Wizard – Professional**

☐ **GCSE Electronics**

☐ **TINA Design Suite V11 Basic (Hobbyist)**
☐ **TINA Design Suite V11 (Student)**

☐ **Teach-In 2**
☐ **Teach-In 3**
☐ **Teach-In 4**
☐ **Teach-In Bundle (3, 4 and 5)**

Full name . . . .
Address . . . .
. . . . Post code . . . . Tel . . . .

Email: . . . .

☐ I enclose cheque/PO in £ sterling payable to Practical Electronics for £ . . . .
☐ Please charge my Visa/Mastercard: £ . . . .

Valid From: . . . . Card expiry date: . . . .

Card No: . . . .

Card Security Code . . . .
(The last 3 digits on or just under the signature strip)

Single License and Site License Versions – overseas readers add £5 to the basic price of each order for airmail postage (**do not add VAT** unless you live in an EU (European Union) country, then add VAT at 20% or provide your official VAT registration number).

Send your order to:
Wimborne Publishing Ltd
113 Lynwood Drive
Wimborne
Dorset BH21 1UU

To order by phone call:

01202 880299. Fax: 01202 843233
Goods are normally sent within seven days

E-mail: stewart.kearn@wimborne.co.uk

Online shop:

www.epemag.com

DIRECT BOOK SERVICE

Teach-In 2017

Introducing the BBC micro:bit

PYTHON CODING ON THE BBC MICRO:BIT

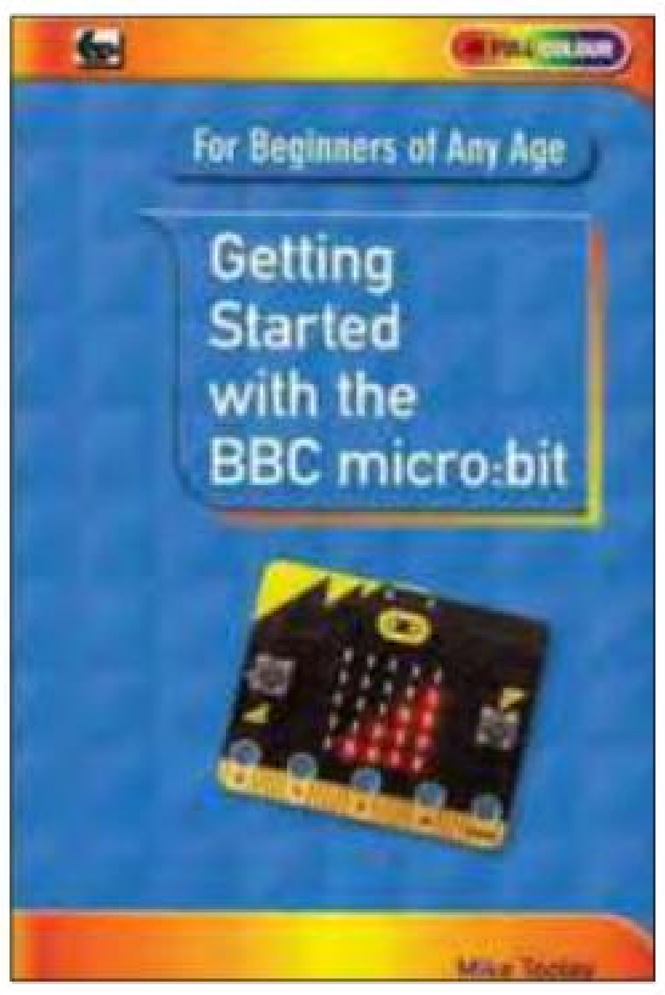
Jim Gatenby

Python is the leading programming language, easy to learn and widely used by professional programmers. This book uses MicroPython, a version of Python adapted for the BBC Micro:bit.

Among the many topics covered are: main features of the BBC micro:bit including a simulation in a web browser screen; various levels of programming languages; Mu Editor for writing, saving and retrieving programs, with sample programs and practice exercises; REPL, an interactive program for quickly testing lines of code; scrolling messages, creating and animating images on the micro:bit's LEDs; playing and creating music, sounds and synthesized speech; using the on-board accelerometer to detect movement of the micro:bit on three axes; glossary of computing terms.

This book is written using plain English, avoids technical jargon wherever possible and covers many of the coding instructions and methods which are common to most programming languages. It should be helpful to beginners of any age, whether planning a career in computing or writing code as an enjoyable hobby.

118 Pages **Order code PYTH MBIT** £7.99



GETTING STARTED WITH THE BBC MICRO:BIT

Mike Tooley

Not just an educational resource for teaching youngsters coding, the BBC micro:bit is a tiny low cost, low-profile ARM-based single-board computer. The board measures 43mm × 52mm but despite its diminutive footprint it has all the features of a fully fledged microcontroller together with a simple LED matrix display, two buttons, an accelerometer and a magnetometer.

Mike Tooley's book will show you how the micro:bit can be used in a wide range of applications from simple domestic gadgets to more complex control systems such as those used for lighting, central heating and security applications. Using Microsoft Code Blocks, the book provides a progressive introduction to coding as well as interfacing with sensors and transducers.

Each chapter concludes with a simple practical project that puts into practice what the reader has learned. The featured projects include an electronic direction finder, frost alarm, reaction tester, battery checker, thermostatic controller and a passive infrared (PIR) security alarm.

No previous coding experience is assumed, making this book ideal for complete beginners as well as those with some previous knowledge. Self-test questions are provided at the end of each chapter, together with answers at the end of the book. So whatever your starting point, this book will take you further along the road to developing and coding your own real-world applications.

108 Pages **Order code BBC MBIT** £7.99

THEORY AND REFERENCE

PRACTICAL ELECTRONICS HANDBOOK – 6th Ed
Ian Sinclair
440 pages **Order code NE21** £33.99

STARTING ELECTRONICS – 4th Ed
Keith Brindley
296 pages **Order code ELSEV100** £18.99

ELECTRONIC CIRCUITS – FUNDAMENTALS & APPLICATIONS – Updated version
Mike Tooley
400 pages **Order code TF43** £32.99

FUNDAMENTAL ELECTRICAL AND ELECTRONIC PRINCIPLES – 3rd Ed
C.R. Robertson
368 pages **Order code TF47** £21.99

A BEGINNER'S GUIDE TO TTL DIGITAL ICs
Robert Penfold
142 pages **OUT OF PRINT BP332** £5.45

UNDERSTANDING ELECTRONIC CONTROL SYSTEMS
Owen Bishop
228 pages **Order code NE35** £36.99

The books listed here have been selected by the *Practical Electronics* editorial staff as being of special interest to everyone involved in electronics and computing. They are supplied by mail order direct to your door.

FOR A FULL DESCRIPTION OF THESE BOOKS AND CD-ROMS SEE THE SHOP ON OUR WEBSITE

www.epemag.com

All prices include UK postage

MICROPROCESSORS

INTERFACING PIC MICROCONTROLLERS – 2nd Ed
Martin Bates
298 pages **Order code NE48** £34.99

PROGRAMMING 16-BIT PIC MICROCONTROLLERS IN C – LEARNING TO FLY THE PIC24
Lucio Di Jasio (Application Segments Manager, Microchip, USA)
496 pages + CD-ROM **Order code NE45** £38.00

INTRODUCTION TO MICROPROCESSORS AND MICROCONTROLLERS – 2nd Ed
John Crisp
222 pages **Order code NE31** £29.99

THE PIC MICROCONTROLLER YOUR PERSONAL INTRODUCTORY COURSE – 3rd Ed
John Morton
270 pages **Order code NE36** £25.00

PIC IN PRACTICE – 2nd Ed
David W. Smith
308 pages **Order code NE39** £24.99

MICROCONTROLLER COOKBOOK
Mike James
240 pages **Order code NE26** £36.99

BOOK ORDERING DETAILS

All prices include UK postage.

For airmail, add £3 per book to Europe, £4 for rest of the world per book.

CD-ROM prices include VAT and/or postage to anywhere in the world.

Send a cheque, (£ sterling only) made payable to: **Practical Electronics** or credit card details (Visa or Mastercard) to:

WIMBORNE PUBLISHING LIMITED,
113 LYNWOOD DRIVE, WIMBORNE, DORSET BH21 1UU

Books are normally sent within seven days of receipt of order.

Please check price (see latest issue of Practical Electronics or website) before ordering from old lists.

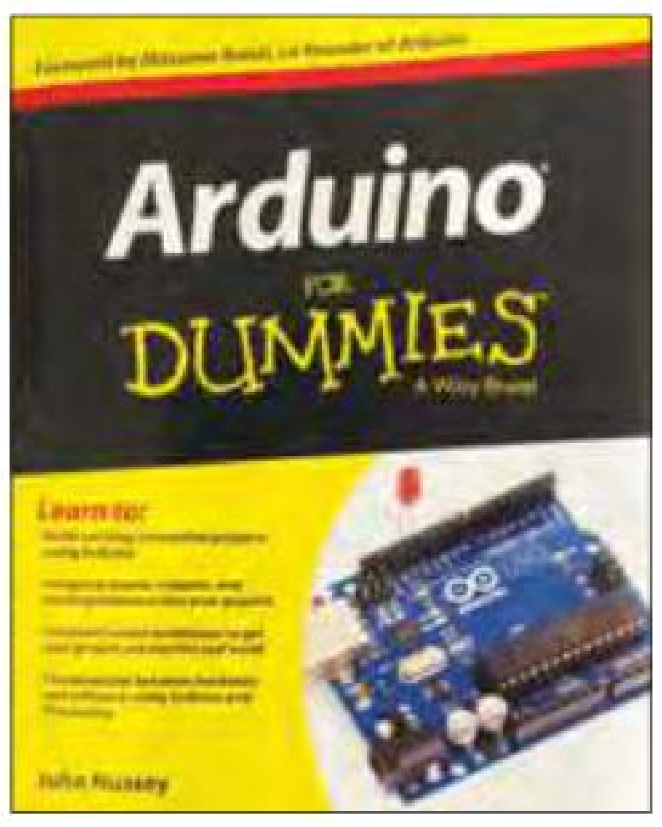
For a full description of these books please see the shop on our website.
Tel: 01202 880299 – E-mail: stewart.kearn@wimborne.co.uk

Order from our online shop at: www.epemag.com

ARDUINO

Teach-In 2016

See opposite for our popular introduction to the Arduino



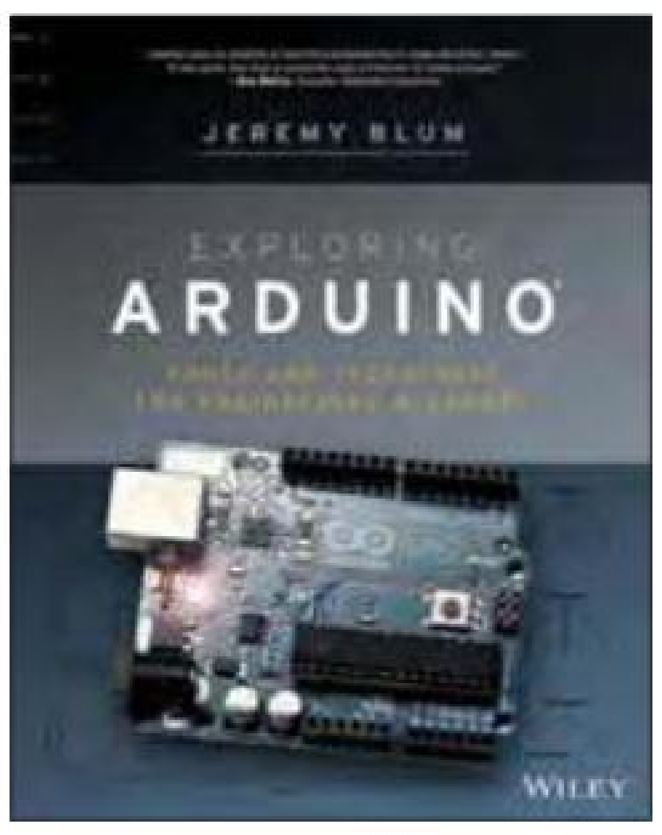
ARDUINO FOR DUMMIES

John Nussey

Arduino is no ordinary circuit board. Whether you're an artist, a designer, a programmer, or a hobbyist, Arduino lets you learn about and play with electronics. You'll discover how to build a variety of circuits that can sense or control real-world objects, prototype your own product, and even create interactive artwork. This handy guide is exactly what you need to build your own Arduino project – what you make is up to you!

- **Learn by doing** – start building circuits and programming your Arduino with a few easy examples – right away!
- **Easy does it** – work through Arduino sketches line by line, and learn how they work and how to write your own.
- **Solder on!** – don't know a soldering iron from a curling iron? No problem! You'll learn the basics and be prototyping in no time.
- **Kitted out** – discover new and interesting hardware to turn your Arduino into anything from a mobile phone to a Geiger counter.
- **Become an Arduino savant** – find out about functions, arrays, libraries, shields and other tools that let you take your Arduino project to the next level
- **Get social** – teach your Arduino to communicate with software running on a computer to link the physical world with the virtual world

438 Pages **Order code ARDDUM01** **£19.99**



EXPLORING ARDUINO

Jeremy Blum

Arduino can take you anywhere. This book is the roadmap.

Exploring Arduino shows how to use the world's most popular microcontroller to create cool, practical, artistic and educational projects. Through lessons in electrical engineering, programming and human-computer interaction, this book walks you through specific, increasingly complex projects, all the while providing best practices that you can apply to your own projects once you've mastered these. You'll acquire valuable skills – and have a whole lot of fun.

- **Explore** the features of commonly used Arduino boards
- **Use** Arduino to control simple tasks or complex electronics
- **Learn** principles of system design, programming and electrical engineering
- **Discover** code snippets, best practices and system schematics you can apply to your original projects
- **Master** skills you can use for engineering endeavours in other fields and with different platforms

357 Pages **Order code EXPARD01** **£26.99**

COMPUTING AND ROBOTICS

NEWNES INTERFACING COMPANION
Tony Fischer-Cripps
295 pages **Order code NE38** **£41.00**

HOW TO BUILD A COMPUTER MADE EASY
Robert Penfold
120 pages **Order code BP707** **£8.49**

EASY PC CASE MODDING
Robert Penfold
192 pages + CD-ROM **Order code BP542** **£8.99**

FREE DOWNLOADS TO PEP-UP AND PROTECT YOUR PC
Robert Penfold
128 pages **Order code BP722** **£7.99**

WINDOWS XP EXPLAINED
N. Kantaris and P.R.M. Oliver
264 pages **Order code BP514** **£7.99**

THE INTERNET – TWEAKS, TIPS AND TRICKS
Robert Penfold
128 pages **Order code BP721** **£7.99**

eBAY – TWEAKS, TIPS AND TRICKS
Robert Penfold
128 pages **Order code BP716** **£7.50**

AN INTRODUCTION TO eBAY FOR THE OLDER GENERATION
Cherry Nixon
120 pages **Order code BP709** **£8.49**

HOW TO FIX YOUR PC PROBLEMS
Robert Penfold
128 pages **Order code BP705** **£8.49**

AN INTRODUCTION TO WINDOWS VISTA
P.R.M. Oliver and N. Kantarris
120 pages **Order code BP703** **£8.49**

WINDOWS 8.1 EXPLAINED
180 Pages **Order code BP747** **£10.99**

AN INTRODUCTION TO THE NEXUS 7
118 Pages **Order code BP744** **£8.99**

COMPUTING FOR THE OLDER GENERATION
Jim Gatenby
308 pages **Order code BP601** **£8.99**

ANDROIDS, ROBOTS AND ANIMATRONS
Second Edition – John Iovine
224 pages **Order code MGH1** **£16.99**

ROBOT BUILDERS COOKBOOK
Owen Bishop
366 pages **Order code NE46** **£26.00**

INTRODUCING ROBOTICS WITH LEGO MINDSTORMS
Robert Penfold
288 pages + **Order code BP901** **£14.99**

MORE ADVANCED ROBOTICS WITH LEGO MINDSTORMS
Robert Penfold
298 pages **Order code BP902** **£14.99**

HOW TO FIX YOUR PC PROBLEMS
Robert Penfold
128 pages **Order code BP705** **£8.49**

WINDOWS 7 – TWEAKS, TIPS AND TRICKS
Andrew Edney
120 pages **Order code BP708** **£8.49**

GETTING STARTED IN COMPUTING FOR THE OLDER GENERATION
Jim Gatenby
120 pages **Order code BP704** **£8.49**

WINDOWS 8.1 EXPLAINED
Noel Kantaris
180 Pages **Order code BP747** **£10.99**

COMPUTING WITH A LAPTOP FOR THE OLDER GENERATION
Robert Penfold
120 pages **Order code BP702** **£8.49**

AN INTRODUCTION TO EXCEL SPREADSHEETS
Jim Gatenby
18 pages **Order code BP701** **£8.49**

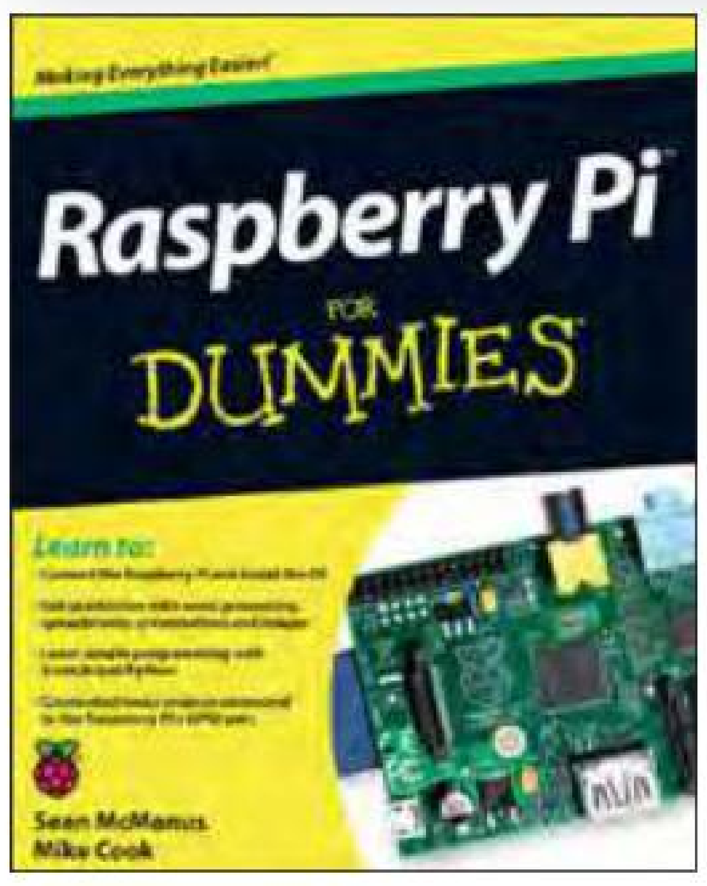
KINDLE FIRE HDX EXPLAINED
118 Pages **Order code BP743** **£8.99**

AUDIO & VIDEO

VALVE AMPLIFIERS – 4th Ed
Morgan Jones
288 pages **Order code ELSEV33** **£46.99**

BUILDING VALVE AMPLIFIERS
Morgan Jones
368 pages **Order code NE40** **£29.00**

RASPBERRY PI



RASPBERRY Pi FOR DUMMIES

Sean McManus and Mike Cook

Write games, compose and play music, even explore electronics – it's easy as Pi! The Raspberry Pi offers a plateful of opportunities, and this great resource guides you step-by-step, from downloading, copying, and installing the software to learning about Linux and finding cool new programs for work, photo editing, and music. You'll discover how to write your own Raspberry Pi programs, create fun games, and much more!

Open this book and find:
What you can do with Python; Ways to use the Raspberry Pi as a productivity tool; How to surf the web and manage files; Secrets of Sonic Pi music programming; A guide to creating animations and arcade games; Fun electronic games you can build; How to build a 3D maze in Minecraft; How to play music and videos on your Raspberry Pi.

400 Pages **Order code RPiDUM01** **£17.99**

RASPBERRY Pi MANUAL: A practical guide to the revolutionary small computer
176 pages **Order code H001** **£17.99**

RASPBERRY Pi USER-GUIDE – 4th Ed
262 pages **Order code JW001** **£20.90**

PROGRAMMING THE RASPBERRY Pi
192 pages **Order code MGH4** **£10.99**

GETTING STARTED WITH RASPBERRY Pi
164 pages **Order code OR01** **£11.50**

TEACH-IN BOOKS

ELECTRONICS TEACH-IN 6

ONLY AVAILABLE ON CD-ROM

ELECTRONICS TEACH-IN 6
A COMPREHENSIVE GUIDE TO RASPBERRY Pi

Mike & Richard Tooley

Teach-In 6 contains an exciting series of articles that provides a complete introduction to the Raspberry Pi, the low-cost computer that has taken the education and computing world by storm.

This latest book in our Teach-In series will appeal to electronic enthusiasts and computer buffs wanting to get to grips with the Raspberry Pi.

Anyone considering what to do with their Pi, or maybe they have an idea for a project but don't know how to turn it into reality, will find Teach-In 6 invaluable. It covers: Programming, Hardware, Communications, Pi Projects, Pi Class, Python Quickstart, Pi World, and Home Baking.

The CD-ROM also contains all the necessary software for the series so that readers can get started quickly and easily with the projects and ideas covered.

160 Pages **Order code ETI6** **£8.99**

ELECTRONICS TEACH-IN 7 (Includes free CD-ROM)

RUNNING LOW – ORDER NOW!

ELECTRONICS TEACH-IN 7
DISCRETE LINEAR CIRCUIT DESIGN

Mike & Richard Tooley

Teach-In 7 is a complete introduction to the design of analogue electronic circuits. Ideal for everyone interested in electronics as a hobby and for those studying technology at schools and colleges. Supplied with a free cover-mounted CD-ROM containing all the circuit software for the course, plus demo CAD software for use with the Teach-In series

Discrete Linear Circuit Design* Understand linear circuit design* Learn with 'TINA' – modern CAD software* Design simple, but elegant circuits* Five projects to build: Pre-amp, Headphone Amp, Tone Control, VU-meter, High Performance Audio Power Amp.

PLUS

Audio Out – an analogue expert's take on specialist circuits
Practically Speaking – the techniques of project building

160 Pages **Order code ETI7** **£8.99**

ELECTRONICS TEACH-IN 8

ONLY AVAILABLE ON CD-ROM

ELECTRONICS TEACH-IN 8
INTRODUCING THE ARDUINO

Mike & Richard Tooley

Hardware – learn about components and circuits; Programming – powerful integrated development system; Microcontrollers – understand control operations; Communications – connect to PCs and other Arduinos

This exciting series has been designed for electronics enthusiasts who want to get to grips with the inexpensive, immensely popular Arduino microcontroller, as well as coding enthusiasts who want to explore hardware and interfacing. Teach-In 8 will provide a one-stop source of ideas and practical information.

The Arduino offers a remarkably effective platform for developing a huge variety of projects; from operating a set of Christmas tree lights to remotely controlling a robotic vehicle through wireless or the Internet. Teach-In 8 is based around a series of practical projects with plenty of information to customise each project.

This book also includes *PIC n' Mix*: PICs and the PICkit 3 - A Beginners guide by Mike O'Keefe and *Circuit Surgery* by Ian Bell - State Machines part 1 and 2.

The CD-ROM includes files for Teach-In 8 plus Microchip MPLAB IDE XC8 8-bit Compiler and PICkit 3 User Guide. Also included is Lab-Nation Smartscope software.

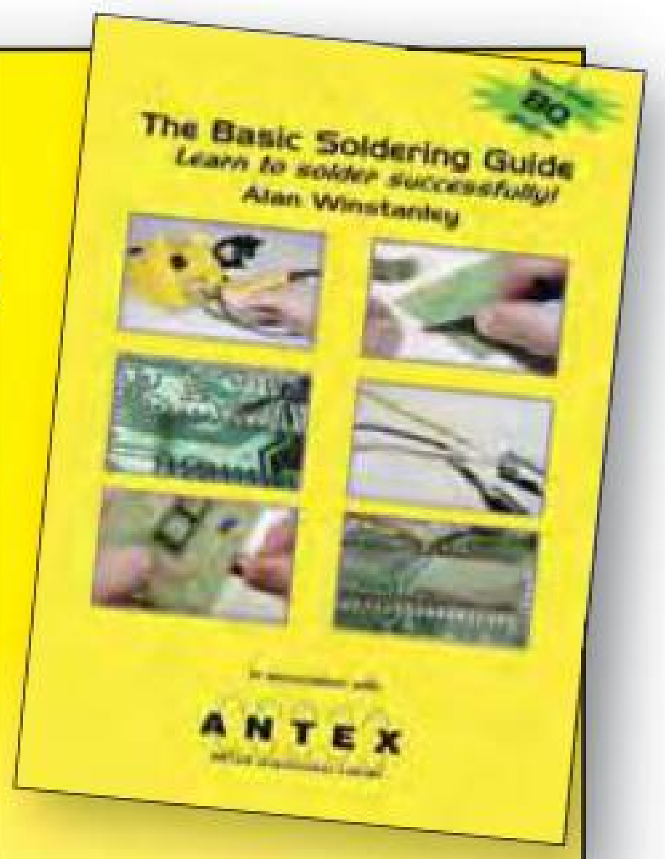
160 Pages **Order code ETI8** **£8.99**

VISIT OUR WEBSITE FOR MORE BOOKS AND FAST, EASY ONLINE ORDERING WWW.EPEMAG.COM

BOOK ORDER FORM

Full name: ....
Address: ....
....
.... Post code: .... Telephone No: ....
Email: ....
☐ I enclose cheque/PO payable to Practical Electronics for £ ....
☐ Please charge my card £ .... Card expiry date..
Card Number ....
Valid From Date .... Card Security Code .... (The last three digits on or just below the signature strip)
Please send book order codes: ....
....
....
Please continue on separate sheet of paper if necessary

THE BASIC SOLDERING GUIDE



LEARN TO SOLDER SUCCESSFULLY!

ALAN WINSTANLEY

The No.1 resource for learning all the basic aspects of electronics soldering by hand.

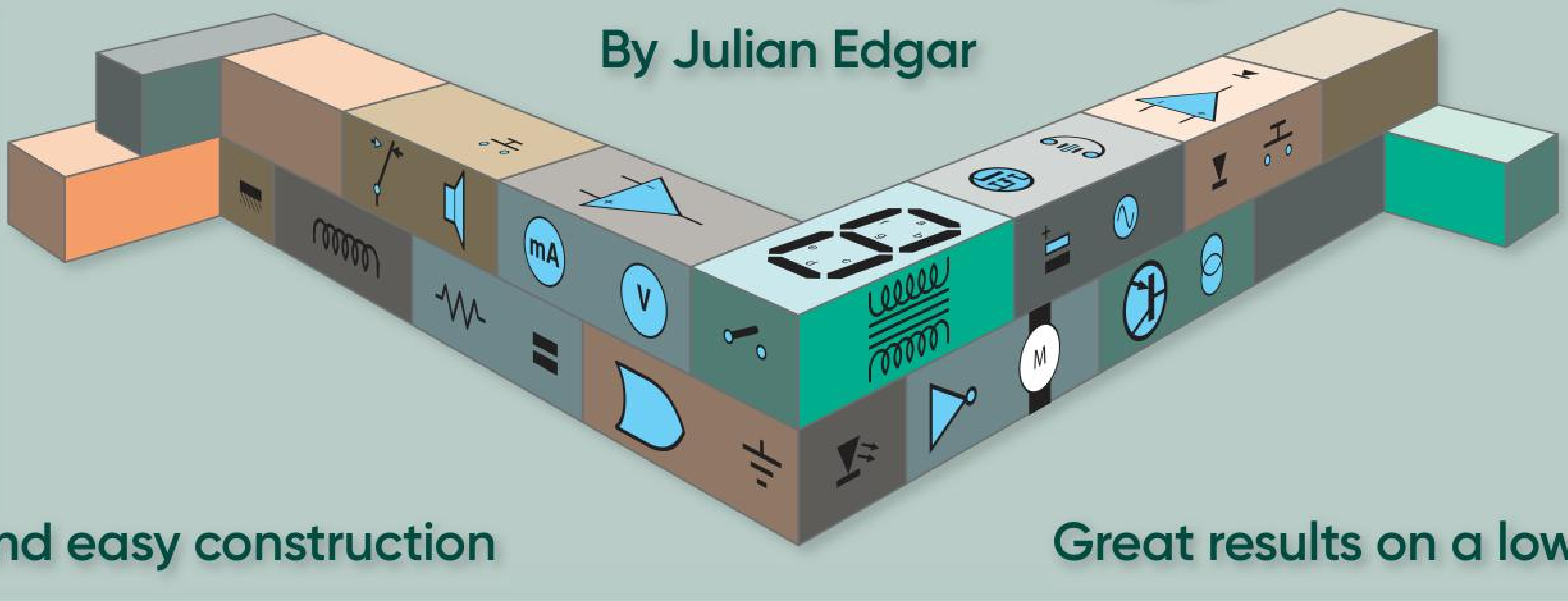
With more than 80 high quality colour photographs, this book explains the correct choice of soldering irons, solder, fluxes and tools. The techniques of how to solder and desolder electronic components are then explained in a clear, friendly and non-technical fashion so you'll be soldering successfully in next to no time! The book also includes sections on reflow soldering and desoldering techniques, potential hazards, useful resources and a very useful troubleshooting guide.

Also ideal for those approaching electronics from other industries, the *Basic Soldering Guide Handbook* is the best resource of its type, and thanks to its excellent colour photography and crystal clear text, the art of soldering can now be learned by everyone!

86 Pages **Order code AW1** **£9.99**

Electronic Building Blocks

By Julian Edgar



Quick and easy construction

Great results on a low budget

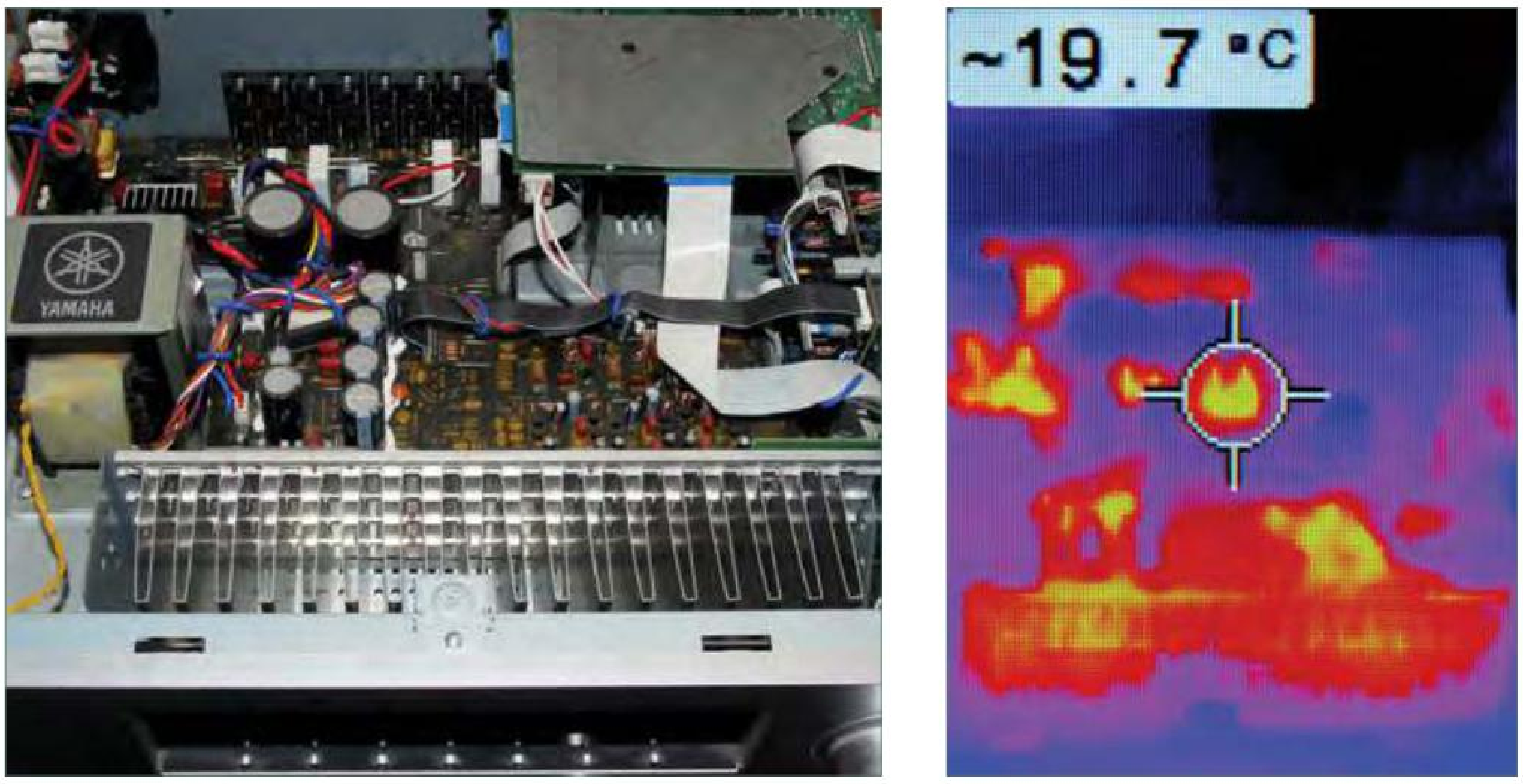
Thermal imaging cameras

The price of thermal imaging cameras continues to plunge – once they cost thousands, now they can be bought for a few hundred. But are the cheaper ones good enough to be useful – and how would you use one? We'll take a look at a 'budget' thermal imaging camera in a moment, but first, something about their uses in electronics.

Thermal imaging

Thermal imaging cameras (or 'thermographic cameras') 'see' infrared radiation and then render it as a visible light image on a colour LCD. Typically, the 'hotter' the colour on the display, the higher the temperature of that area.

Cameras vary in the resolution of the sensor and the temperature range they can sense – in both cases, more equals pricier. Cameras are also available that mix a visible image with the infrared image (allowing you to more easily work out what the camera is seeing), and most cameras store images on a memory card for later download.



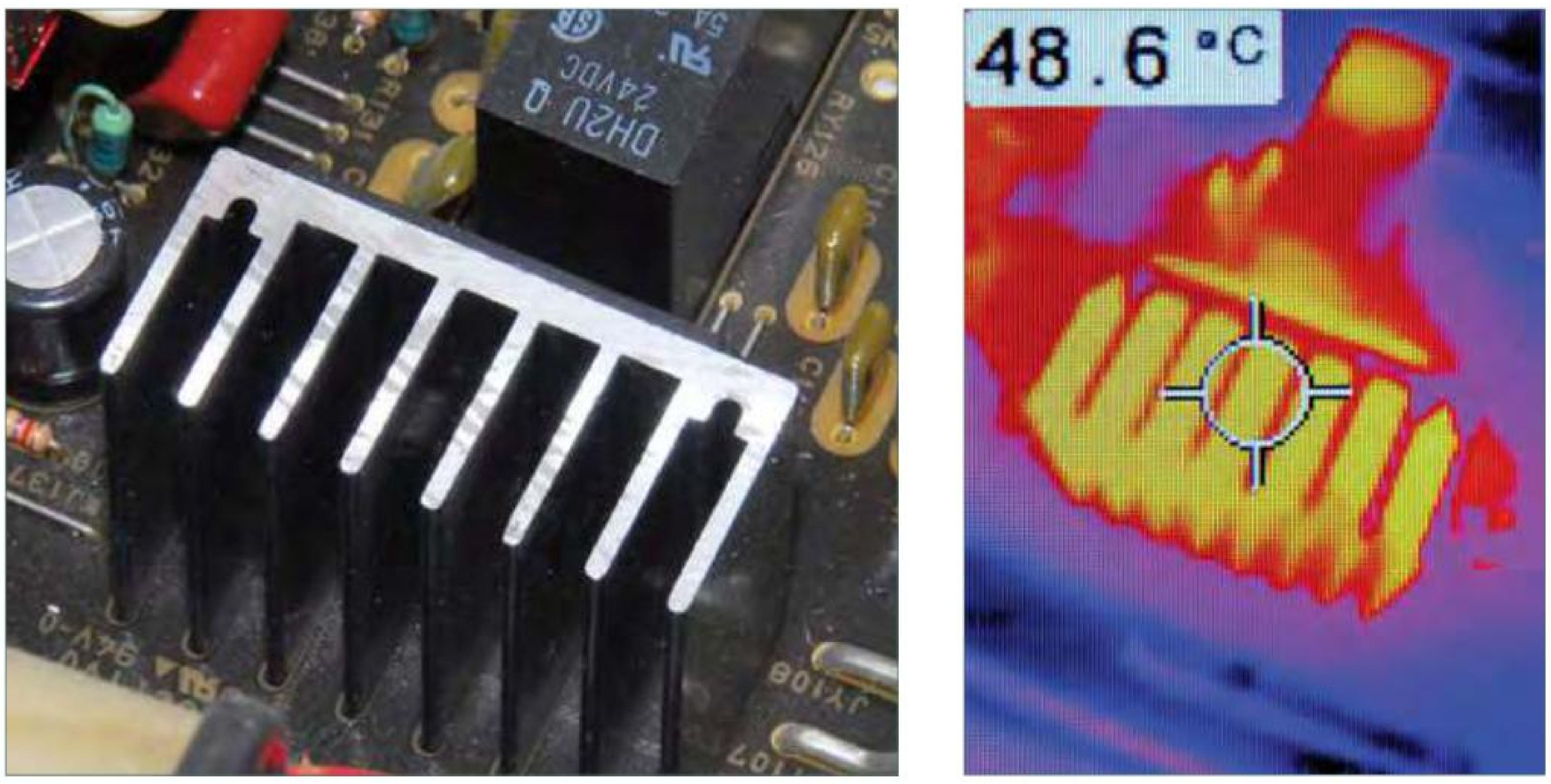
The inside of an amplifier photographed with a conventional camera (left) and with the Flir TG-130 (right). It's easy to see at a glance where heat is being dissipated and then move the thermal camera in for a closer look.

Uses of the camera

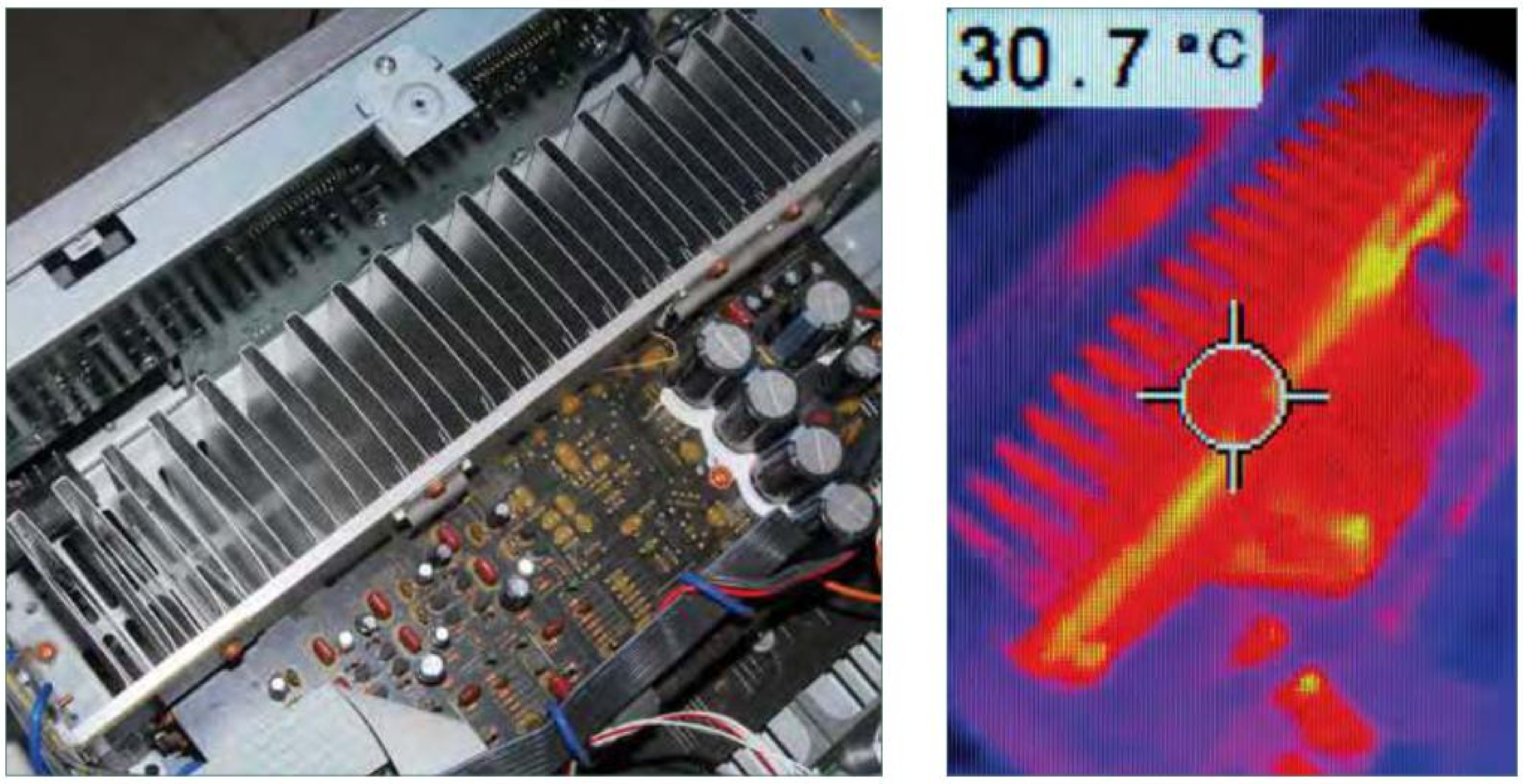
So why would you want a thermal imaging camera – especially over an infrared thermometer? Both can remotely sense temperature, and the infrared thermometer is a lot cheaper. Well, thermal imaging cameras have a couple of distinct advantages. The biggest advantage is that you can quickly scan whole areas. For example, if you are looking at a switchboard, or the insides of a piece of electronic equipment, you can see at a glance if there are any hot spots. To do the same with an infrared thermometer

FLIR
FLIR
FLIR

The TG-130 camera is the entry thermal imaging camera from Flir. It has a resolution of 80 x 60 pixels and a minimum focussing distance of 100mm. Images cannot be stored – one of the downsides associated with its relatively low cost.



A heatsink in the amplifier's power supply, as photographed with a normal camera (left) and the Flir TG-130 (right). The heatsink is running at nearly 50°C in 20°C ambient conditions – that is getting rather too hot.



Main amplifier heatsink: thermal image shows the heatsink working well, with an even temperature distribution on its base and a temperature rise of only 10°C over ambient.

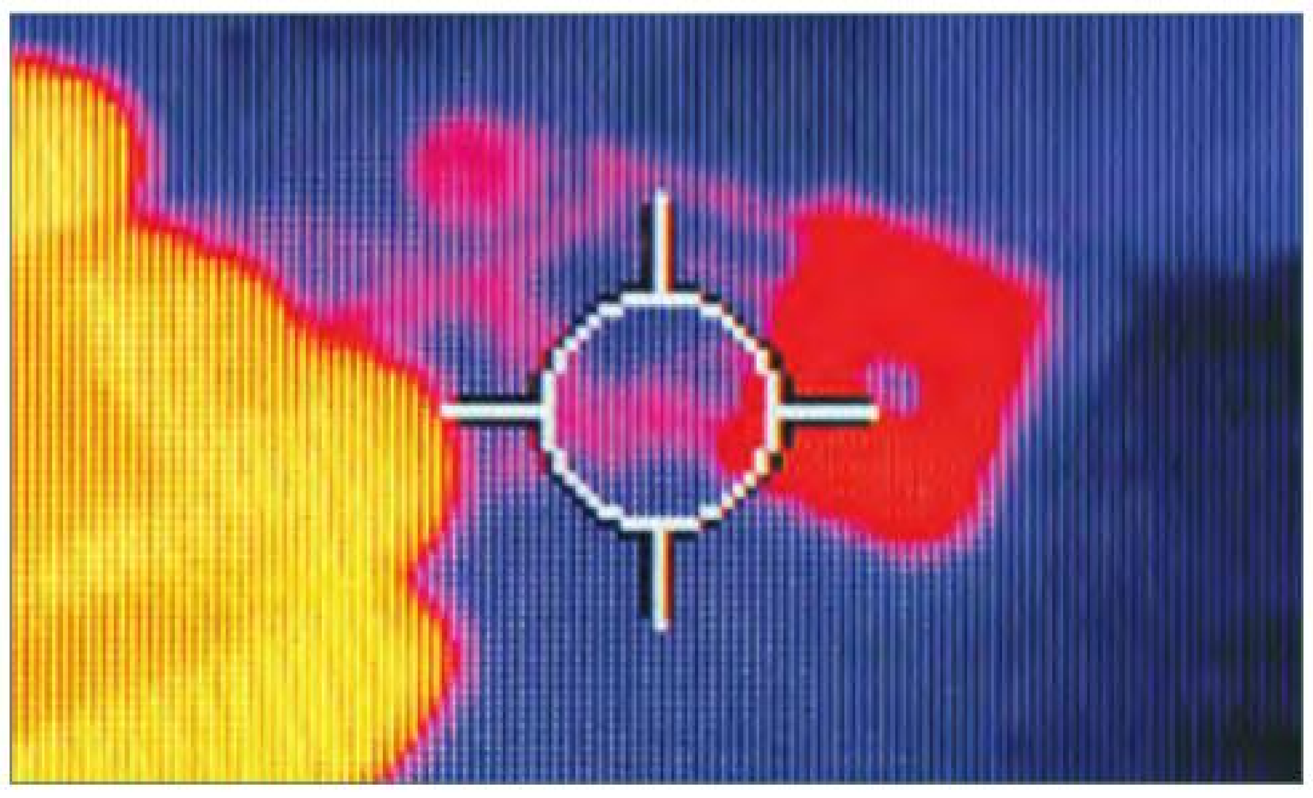
would require that you methodically scanned each square centimetre to see all the different temperatures. This would take a lot longer and it would also be very easy to miss a spot.

Another advantage is that thermal imaging cameras adjust the scale that they are using, depending on the variation in temperature. For example, if you scan the inside of a heated room in winter, you can immediately see the hot air gathered near the ceiling – despite the fact that the hot air might only be a few degrees warmer than air nearer the floor. This means that temperature differences are immediately obvious – even though the magnitude of those differences may be small.

Finally, after several months of ownerships, I found myself using the thermal imaging camera in applications I'd never even previously considered. 'Seeing' timber studs inside walls, being amazed at how much heat was escaping a room through the aluminium window frames, and testing the insulating properties of a sleeping bag (put a person inside, and you can easily see where the heat is leaking through) – to name just a few uses.

And electronics-specific uses? Obviously, the infrared camera is most useful in applications where you are trying to dissipate heat (ie heatsinks) or want to ensue no heat is being generated.

Taking the second application first, when developing a generator-powered bicycle lighting system, I intended to pot the electronics in epoxy. To ensure the rectifier wouldn't become overly warm with no airflow, I tested the bare system at full load and then used the thermal camera to ensure the temperature rise of the rectifier was low – it was. In this situation, even placing fingers on the small component would have been enough to transfer detectable heat to it.



Testing the temperature increase of a bridge rectifier that was to be later potted in epoxy. The rectifier is the rectangle with the hole in it at right – its relative coolness can be compared with fingers (left).

And the other application – getting rid of heat? I have a Yamaha HTR-5840 amplifier I use as a general-purpose speaker-testing amp. (It has surprisingly good specifications for an amplifier than can now be bought quite cheaply.) The amplifier uses a large but rather basic-looking heatsink, and has a few smaller heatsinks scattered around (eg in the power supply). I often drive the amplifier hard, but I previously had no idea how hot anything was getting. It's pretty hard to tell just by looking, although I did notice a bridge rectifier without a heatsink that looked like it could be a 'hot' culprit.

To see what was actually happening, I ran the amplifier at high load for about an hour, removed the cover and then used the thermal imaging camera. This showed that the main heatsink was working fine; the bridge rectifier I'd noticed earlier didn't in fact need a heatsink, but another relatively small heatsink in the power supply was running at nearly 50°C in 20°C ambient conditions. This was clearly the heatsink that needed improving.

Entry-level thermal camera

So which thermal imaging camera can be used in this way? The Flir TG-130 is one of the cheapest models available from a reputable maker. It has a field of view of 55 x 43 degrees and a minimum focussing distance of 100mm. The display is a 1.8-inch TFT LCD and the sensor has a resolution of 80 × 60 pixels. The sensed temperature range is –10°C to +150°C (quite a low maximum compared with some thermal imaging cameras; one of the trade-offs for its lower cost.) The camera also does not overlay a visible light image over the infrared image – you get only the temperature display. It doesn't feel particularly sturdy but, apparently it is rated to survive a 2m drop. It uses three AAA cells and has a claimed battery life of four hours of continuous scanning.

Victor, the pet sheep, on a cold morning!

Using the camera is very simple. After the power button has been pushed, the instrument takes about four seconds to boot up. Then it's just a case of pointing it at what you are interested in and pulling the trigger button, which freezes the image on the LCD until the trigger is again pulled.

Displayed on the image is a circle within cross hairs, and the temperature within this target is indicated by a digital display within the screen.

Resolution issues?

My biggest concern when buying the instrument was the relatively low resolution – after all, a visible light image with only 80 × 60 pixels doesn't look very good! In use, however, I found this resolution posed surprisingly few issues – in part because of the ability of the instrument to get within 100mm of whatever you are measuring. By physically 'zooming in', you can get a very good idea of what is hot or cold. Having said that, no doubt in some situations having a higher resolution would be an advantage, but the Flir is quite usable. The displayed temperature is fairly accurate – within 5°C. (As with all infrared thermometers, accuracy depends on the emissivity of the surface versus the calibration of the meter. The Flir TG-130 cannot be calibrated for different emissivities, meaning that a black object will read differently to a silver object – even when they are the same temperature.)

At the time of writing, the TG-130 costs about £200 – not cheap! However, with its numerous uses around the home and workplace, it's a tool where once you part with the cash, you'll find use after use for it.

DISCLAIMER – the Flir TG-130 tested here was purchased by the reviewer without discount, support or any other incentive from the manufacturer.

Practical Electronics PCB SERVICE

PROJECT	CODE	PRICE
JUNE 2017		
Ultrasonic Garage Parking Assistant	07102122	£10.45
Hotel Safe Alarm	03106161	£8.00
100dB Stereo LED Audio Level/VU Meter	01104161	£17.75
JULY 2017		
Micromite-Based Super Clock	07102122	£10.45
Brownout Protector for Induction Motors	10107161	£12.90
AUGUST 2017		
Micromite-Based Touch-screen Boat Computer GPS	07102122	£10.45
Fridge/Freezer Alarm	03104161	£8.00
SEPTEMBER 2017		
Compact 8-Digit Frequency Meter	04105161	£12.88
NOVEMBER 2017		
50A Battery Charger Controller	11111161	£12.88
Micropower LED Flasher (45 × 47mm)	16109161	£8.00
(36 × 13mm)	16109162	£5.60
Phono Input Converter	01111161	£8.00
DECEMBER 2017		
Precision Voltage and Current Reference – Part 2	04110161	£15.35
JANUARY 2018		
High-Power DC Motor Speed Controller – Part 1	11112161	£12.88
Build the SC200 Amplifier Module	01108161	£12.88
FEBRUARY 2018		
GPS-Syncronised Analogue Clock Driver	04202171	£12.88
High-Power DC Motor Speed Controller – Part 2		
+ Control Board	11112161	£12.88
+ Power Board	11112162	£15.30
MARCH 2018		
Stationmaster Main Board	09103171	£17.75
+ Controller Board	09103172	
Build the SC200 Amplifier Module – Power Supply	01109111	£16.45
APRIL 2018		
Spring Reverberation Unit	01104171	£15.30
DDS Sig Gen Lid	Black	£8.05
DDS Sig Gen Lid	Blue	£7.05
DDS Sig Gen Lid	Clear	£8.05
MAY 2018		
High Performance RF Prescaler	04112162	£10.45
Micromite BackPack V2	07104171	£10.45
Microbridge	24104171	£5.60
JUNE 2018		
High Performance 10-Octave Stereo Graphic Equaliser	01105171	£15.30
JULY 2018		
Touchscreen Appliance Energy Meter – Part 1	04116061	£17.75
Automotive Sensor Modifier	05111161	£12.88
AUGUST 2018		
Universal Temperature Alarm	03105161	£7.05
Power Supply For Battery-Operated Valve Radios	18108171 18108172 18108173 18108174	£27.50
SEPTEMBER 2018		
3-Way Active Crossover	01108171	£22.60
Ultra-low-voltage Mini LED Flasher	16110161	£5.60

PROJECT	CODE	PRICE
OCTOBER 2018		
6GHz+ Touchscreen Frequency Counter	04110171	£12.88
Two 230VAC MainsTimers	10108161 10108162	£12.88
NOVEMBER 2018		
Super-7 AM Radio Receiver	06111171	£27.50
FEBRUARY 2019		
1.5kW Induction Motor Speed Controller	10105122	£35.00
MARCH 2019		
10-LED Bargraph Main Board	04101181	£11.25
+Processing Board	04101182	£8.60
APRIL 2019		
Heater Controller	10104181	£14.00
MAY 2019		
2× 12V Battery Balancer	14106181	£5.60
Deluxe Frequency Switch	05104181	£10.45
USB Port Protector	07105181	£5.60
JUNE 2019		
Arduino-based LC Meter	04106181	£8.00
USB Flexitimer	19106181	£10.45

PE/EPE PCB SERVICE

Order Code **Project** **Quantity** **Price**

. . . .

. . . .

. . . .

. . . .

. . . .

Name . . . .
Address . . . .
. . . .
Tel . . . .
Email . . . .
I enclose payment of £ . . . . (cheque/PO in **£ sterling only**)
payable to: **Practical Electronics**

Card No . . . .
Valid From . . . . Expiry Date . . . .
Card Security No . . . .

You can also order PCBs by phone, Fax, Email or via the shop on our website on a secure server: www.epemag.com

PCBs for most recent *PE*/*EPE* constructional projects are available. From the July 2013 issue onwards, PCBs with eight-digit codes have silk screen overlays and, where applicable, are double-sided, plated-through hole, with solder mask. They are similar to photos in the project articles. Earlier PCBs are likely to be more basic and may not include silk screen overlay, be single-sided, lack plated-through holes and solder mask.

Always check price and availability in the latest issue or online. A large number of older boards are listed for ordering on our website.

We do not supply kits or components for our projects. For older projects it is important to check the availability of all components before purchasing PCBs.

Back issues of articles are available – see Back Issues page for details.

All prices include VAT and UK p&p. Add £2 per board for airmail outside of Europe. Orders and payment should be sent to:

Practical Electronics, Wimborne Publishing Ltd
113 Lynwood Drive, Merley, Wimborne, Dorset BH21 1UU
Tel 01202 880299 Fax 01202 843233
Email: stewart.kearn@epemag.wimborne.co.uk
On-line Shop: www.epemag.com

Cheques should be made payable to Practical Electronics **(Payment in £ sterling only)**.

NOTE: While 95% of our boards are held in stock and are dispatched within seven days of receipt of order, please allow a maximum of 28 days for delivery if we need to restock.

CLASSIFIED ADVERTISING

Practical Electronics

If you want your advertisements to be seen by the largest readership at the most economical price our classified page offers excellent value. The rate for semi-display space is £10 (+VAT) per centimetre high, with a minimum height of 2·5cm. All semi-display adverts have a width of 5.5cm. The prepaid rate for classified adverts is 40p (+VAT) per word (minimum 12 words).

Cheques are made payable to Practical Electronics. **VAT must be added**. Advertisements with remittance should be sent to: Practical Electronics, 113 Lynwood Drive, Wimborne, Dorset, BH21 1UU.

Tel 01202 880299 Email: stewart.kearn@wimborne.co.uk

For rates and further information on display and classified advertising please contact our Advertisement Manager, Stewart Kearn – see below.

Practical Electronics reaches more UK readers than any other UK monthly hobby electronics magazine. Our sales figures prove it.

We have been the leading monthly magazine in this market for the last twenty-seven years.

BOWOOD ELECTRONICS LTD
Suppliers of Electronic Components
www.bowood-electronics.co.uk
Unit 10, Boythorpe Business Park, Dock Walk, Chesterfield, Derbyshire S40 2QR. Sales: 01246 200 222
Send large letter stamp for Catalogue

The British Amateur Electronic Club at:
baec.tripod.com
Has many interesting articles on computers; digital electronics and analogue electronics.

COAST ELECTRONICS
BREAKOUTS-COMPONENTS-CONTRACT DESIGN-3D PRINTER PARTS-MUSICAL-MICROCONTROLLERS
WWW.COASTELECTRONICS.CO.UK

MISCELLANEOUS

PIC DEVELOPMENT KITS, DTMF kits and modules, CTCSS Encoder and Decoder/Display kits. Visit www.cstech.co.uk

VALVES AND ALLIED COMPONENTS IN STOCK. Phone for free list. Valves, books and magazines wanted. Geoff Davies (Radio), tel. 01788 574774.

Andrew Kenny – Qualified Patent Agent
EPO UKIPO USPTO
Circuits Electric Machinery Mechatronics
Web: www.akennypatentm.com
Email: Enquiries@akennypatentm.com
Tel: 0789 606 9725

ADVERTISING INDEX

CRICKLEWOOD ELECTRONICS 30
ESR ELECTRONIC COMPONENTS 30
HAMMOND ELECTRONICS Ltd 8
JPG ELECTRONICS 80
LASER BUSINESS SYSTEMS 22
MICROCHIP. . . . . .Cover (ii), Cover (iii) & 5
PEAK ELECTRONIC DESIGN. . . . . Cover (iv)
PICO TECHNOLOGY. 11
POLABS D.O.O.. 35
QUASAR ELECTRONICS 2
SOUNDTRONICS 3
STEWART OF READING. 9
TAG-CONNECT 22

Advertisement offices

Wimborne Publishing Ltd
113 Lynwood Drive
Merley
Wimborne,
Dorset BH21 1UU

Tel 01202 880299
Fax 01202 843233
Email stewart.kearn@wimborne.co.uk
Web www.epemag.com

For editorial contact details see page 7.

Next Month – in the July issue

Full-wave 10A Universal Motor Speed Controller

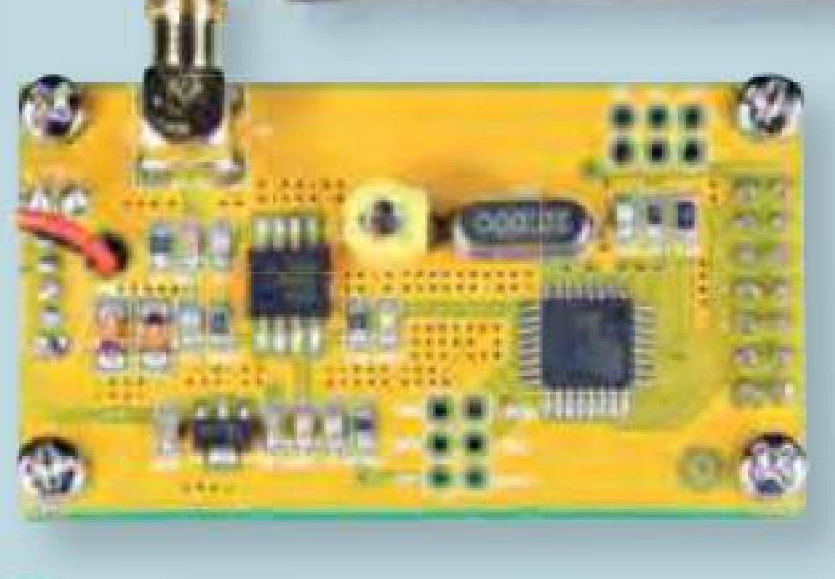
A simple but highly effective triac-based circuit giving smooth, full-range speed control for electric drills, lawn edgers, circular saws, routers or any other appliance with universal motors (ie, brush-type), rated up to 10A.

Recurring Event Reminder

This simple circuit will remind you to perform any regular task, which needs to be done every few hours, days, weeks or fortnights. It flashes an LED, and you can even connect a piezo siren.

Temperature Switch Mk2

Turn on a pump or fan if something is too hot… or turn on a heater if it's too cold. Two sets of changeover contacts allow a flexible switching arrangement.

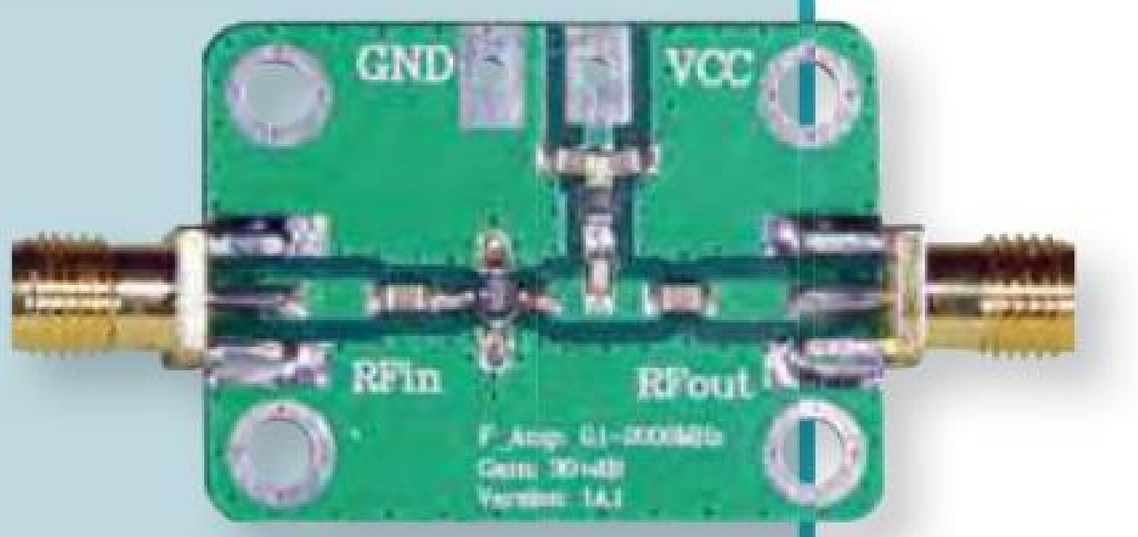


Low-cost Electronic Modules – Part 17

Two modules: a tiny digital counter module which operates to 500MHz; plus a wideband amplifier. The two can combine to make a sensitive frequency counter.

Teach-In 2019 – Part 8

Next month, we will bring this series to a close with a look at battery types and characteristics. Our *Practical Project* will be a handy battery monitor.



MIDI Ultimate Synthesiser – Part 6

We will cover building the case, final assembly, connections to the jack sockets and the synth's final calibration.

PLUS!

All your favourite regular columns from *Cool Beans*, *Make it with Micromite*, *Circuit Surgery* and *Electronic Building Blocks* to *PIC n' Mix*, *Audio Out*, *Techo Talk* and *Net Work*.

On sale 6 June 2019

Content may be subject to change

WELCOME TO JPG ELECTRONICS
Selling Electronics in Chesterfield for 29 Years

Open Monday to Friday 9am to 5:30pm
And Saturday 9:30am to 5pm

- Aerials, Satellite Dishes & LCD Brackets
- Audio Adaptors, Connectors & Leads
- BT, Broadband, Network & USB Leads
- Computer Memory, Hard Drives & Parts
- DJ Equipment, Lighting & Supplies
- Extensive Electronic Components
  - *ICs, Project Boxes, Relays & Resistors*
- Raspberry Pi & Arduino Products
- Replacement Laptop Power Supplies
- Batteries, Fuses, Glue, Tools & Lots more…

T: 01246 211 202
E: sales@jpgelectronics.com
JPG Electronics, Shaw's Row,
Old Road, Chesterfield, S40 2RB
W: www.jpgelectronics.com

JPG Electronics

Retail & Trade Welcome • Free Parking • Google St View Tour: S40 2RB

Calling all subscribers!

Practical Electronics

If you are one of our valued subscribers then please note that we have changed the way we send subscription renewal reminders.

Instead of sending you a renewal card, we now print a box on the address sheet that comes with your copy of Practical Electronics.

This box will advise you of the last issue of your current subscription.

To renew, you have three easy choices:

1. Call us on: 01202 880299
2. Visit our website at: www.epemag.com
3. Send a cheque with your details to:

Electron Publishing Limited
113 Lynwood Drive
Wimborne
Dorset BH21 1UU

Published on approximately the first Thursday of each month by Electron Publishing Limited, 1 Buckingham Road, Brighton, East Sussex BN1 3RA. Printed in England by Acorn Web Offset Ltd., Normanton WF6 1TW. Distributed by Seymour, 86 Newman St., London W1T 3EX. Subscriptions UK: £25.00 (6 months); £47.00 (12 months); £89.00 (2 years). EUROPE: airmail service, £30.00 (6 months); £56.00 (12 months); £107.00 (2 years). REST OF THE WORLD: airmail service, £37.00 (6 months); £70.00 (12 months); £135.00 (2 years). Payments payable to 'Practical Electronics', Electron Publishing Limited, 113 Lynwood Drive, Wimborne, Dorset BH21 1UU. Email: stewart.kearn@epemag.wimborne.co.uk. PRACTICAL ELECTRONICS is sold subject to the following conditions, namely that it shall not, without the written consent of the Publishers first having been given, be lent, resold, hired out or otherwise disposed of by way of Trade at more than the recommended selling price shown on the cover, and that it shall not be lent, resold, hired out or otherwise disposed of in a mutilated condition or in any unauthorised cover by way of Trade or affixed to or as part of any publication or advertising, literary or pictorial matter whatsoever.

Faster Programming, Wider Voltage Ranges and Enhanced Interface Options

PICkit™ 4 enables faster design development

MPLAB®
PICkit™ 4
IN-CIRCUIT DEBUGGER
MICROCHIP

With five times faster programming and a wider 1.2V to 5V range, the Microchip MPLAB® PICkit™ 4 Development Tool supports low-cost development of even more Microchip controllers.

PICkit™ 4 also introduces improved USB connectivity with a hi-speed USB 2.0 interface and advanced debugging over 4-wire JTAG and serial wire debug with streaming data gateway in addition to legacy interfaces.

The new advanced interfaces support in-circuit programming and debugging of Microchip's CEC1702 hardware cryptography-enabled MCUs, in addition to dsPIC® Digital Signal Controllers and PIC® microcontrollers from 8- to 32-bit.

MICROCHIP

www.microchip.com/PICkit4eu

The Microchip name and logo, the Microchip logo, MPLAB, PIC and dsPIC are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries. PICkit and In-Circuit Serial Programming (ISCP) are trademarks of Microchip Technology Inc. in the U.S.A. and other countries. All other trademarks mentioned herein are the property of their respective companies. © 2018 Microchip Technology Inc. All rights reserved.
DS-50002745A. MEC2206Eng04/18

PEAK®
electronic design ltd

Celebrating the return of Practical Electronics!

Limited time offer only

Use code PE10 during checkout
(or on the phone!)

DCA75 Advanced Semiconductor Analyser and Curve Tracer

DCA75
atlas DCA Pro
USB + Graphics
Vbe=1.216V
at Ib=3.6mA
IcLeak=0.000mA
PEAK®
electronic design ltd
on - test
scroll - off
warning: avoid charge/voltages

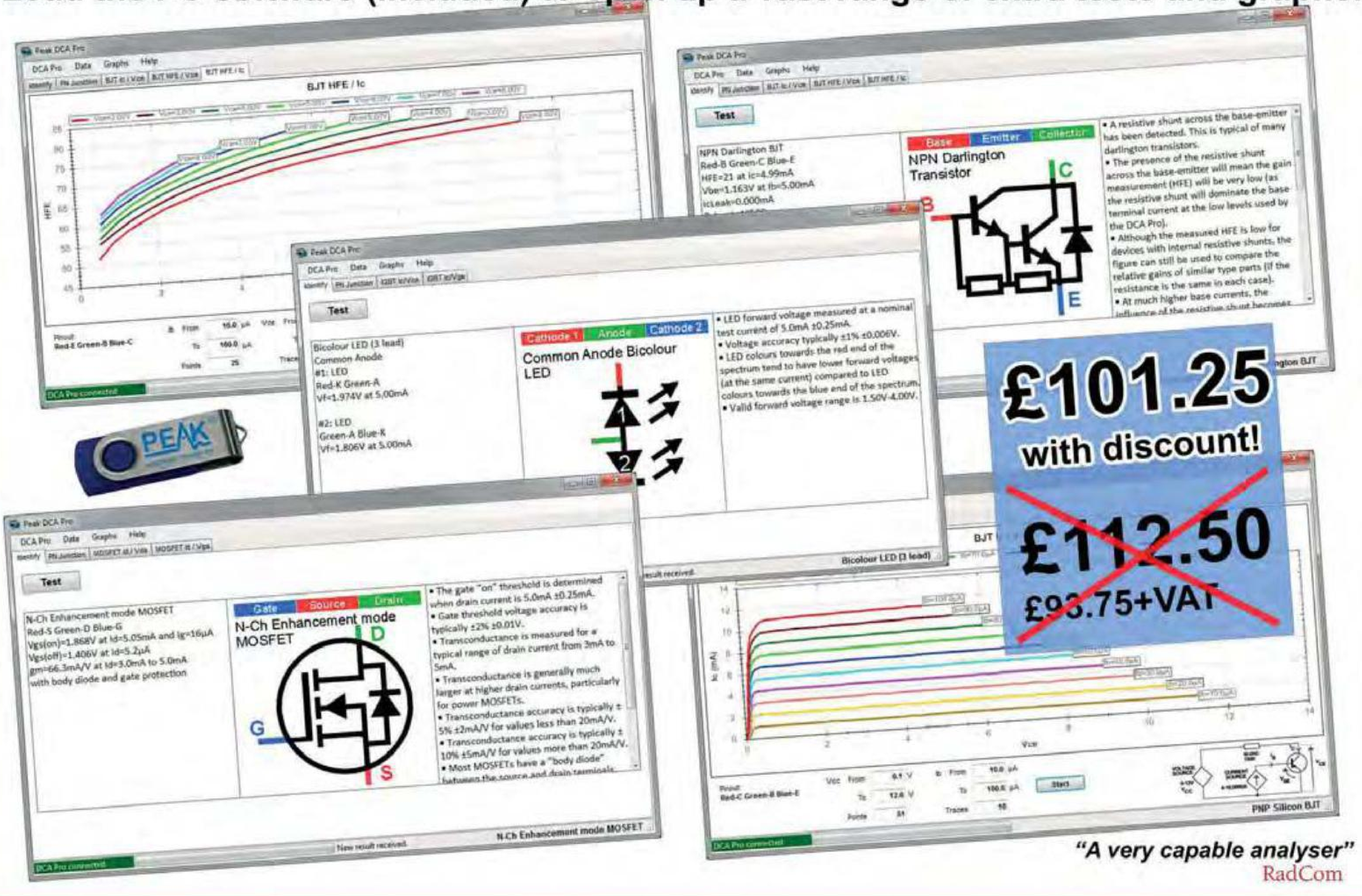
A versatile instrument that you'll love using for testing a huge range of semiconductor components.

Supports MOSFETs, transistors, diodes, LEDs, JFETs, IGBTs and more.

Connect any way round and let the ***DCA Pro*** identify the component type, pinout and lots of parameters.

Can be used standalone or connected to a Windows® PC (XP up to Win10).

Load the PC software (included) to open up a vast range of extra tests and graphs.


Peak DCA Pro
DCA Pro
Data
Graphs
Help
Identify
PN Junction
BJT Ic / Vce
BJT HFE / Vce
BJT HFE / Ic
BJT HFE / Ic
HFE
Test
Base
Emitter
Collector
NPN Darlington Transistor
B
C
E
IGBT Ic/Vce
IGBT Ic/Vge
Cathode 1
Anode
Cathode 2
Common Anode Bicolour LED
1
2
MOSFET Id / Vds
MOSFET Id / Vgs
Gate
Source
Drain
N-Ch Enhancement mode MOSFET
G
D
S
PEAK
Vce
Ic (mA)
PNP Silicon BJT
DCA Pro connected

NPN Darlington BJT
Red-B Green-C Blue-E
HFE=21 at Ic=4.99mA
Vbe=1.163V at Ib=5.00mA
IcLeak=0.000mA

- A resistive shunt across the base-emitter has been detected. This is typical of many darlington transistors.
- The presence of the resistive shunt across the base-emitter will mean the gain measurement (HFE) will be very low (as the resistive shunt will dominate the base terminal current at the low levels used by the DCA Pro).
- Although the measured HFE is low for devices with internal resistive shunts, the figure can still be used to compare the relative gains of similar type parts (if the resistance is the same in each case).
- At much higher base currents, the influence of the resistive shunt becomes

Bicolour LED (3 lead)
Common Anode
#1: LED
Red-K Green-A
Vf=1.974V at 5.00mA

#2: LED
Green-A Blue-K
Vf=1.806V at 5.00mA

- LED forward voltage measured at a nominal test current of 5.0mA ±0.25mA.
- Voltage accuracy typically ±1% ±0.006V.
- LED colours towards the red end of the spectrum tend to have lower forward voltages (at the same current) compared to LED colours towards the blue end of the spectrum.
- Valid forward voltage range is 1.50V-4.00V.

N-Ch Enhancement mode MOSFET
Red-S Green-D Blue-G
Vgs(on)=1.868V at Id=5.05mA and Ig=16µA
Vgs(off)=1.406V at Id=5.2µA
gm=66.3mA/V at Id=3.0mA to 5.0mA
with body diode and gate protection

- The gate "on" threshold is determined when drain current is 5.0mA ±0.25mA.
- Gate threshold voltage accuracy is typically ±2% ±0.01V.
- Transconductance is measured for a typical range of drain current from 3mA to 5mA.
- Transconductance is generally much larger at higher drain currents, particularly for power MOSFETs.
- Transconductance accuracy is typically ± 5% ±2mA/V for values less than 20mA/V.
- Transconductance accuracy is typically ± 10% ±5mA/V for values more than 20mA/V.
- Most MOSFETs have a "body diode" between the source and drain terminals.

New result received.

£101.25
with discount!
~~£112.50~~
~~£93.75+VAT~~

"A very capable analyser"
RadCom

It's only possible to show summary specifications here. Please ask if you'd like detailed data. Further information is also available on our website. Product price refunded if you're not happy.

Tel. 01298 70012
www.peakelec.co.uk
sales@peakelec.co.uk

Atlas House, 2 Kiln Lane
Harpur Hill Business Park
Buxton, Derbyshire
SK17 9JL, UK

UK designed **and** manufactured